Actes de la conférence CAID 2023

(Conference on Artificial Intelligence for Defense)

Organisée par

DGA

MINISTÈRE DES ARMÉES
Liberté
Égalité
Fraternité

L'IA appliquée aux modèles physiques

Vers la simulation de composants radio-fréquence par des réseaux de neurones contraint par la physique
Towards simulation of radio-frequency component with physics informed neural networks
Mathieu Riou, Mouhamed Coulibaly and Jean-Pierre Marcy

Approche Bayésienne pour l'Estimation des Paramètres de la Dynamique Latérale d'un Véhicule
Bayesian Approach for Estimating Parameters of Vehicle Lateral Dynamics
Fabien Lionti, Nicolas Gutowski, Sébastien Aubin and Philippe Martinet

L'IA Appliquée à la Cybersécurité

Application du chiffrement fonctionnel sur données confidentielles pour la conception de modèles d'apprentissage automatique
An example of functional encryption for machine learning on private data
Tom Laborde, Alexandre Gensse, Phillipe Chartier, Mohammed Lemou, Florian Méhats, Fabien Chaillan and Clément Gicquel

Vers une définition de la Guerre Cognitive
Towards a Definition of Cognitive Warfare
Marie Morelle, Damien Marion, Julien Cegarra and Jean-Marc André

Red Team LLM : vers une solution automatique, adaptative et robuste
Red Team LLM: towards an adaptive and robust automation solution
Christophe Genevey-Metat, Dorian Bachelot, Tudy Gourmelen, Adrien Quemat, Pierre-Marie Satre, Loïc Scotto Di Perrotolo, Maximilien Chaux, Pierre Delesques and Olivier Gesny

La recherche sémantique sur données de défense: Générer du texte pour mesurer la robustesse
Semantic search for defence application: using AI text-generation to measure robustness
Bruno Carron, Claude Fendzi and Guillaume Gadek

Un système d'apprentissage ensembliste explicable pour détecter des attaques réseau inconnues
An explainable-by-design ensemble learning system to detect unknown network attacks
Céline Minh, Kevin Vermeulen, Cédric Lefebvre, Philippe Owezarski and William Ritchie

IA & facteur humain

Confiance dans l'Automatisation : Analyser et Modéliser la Confiance Longitudinale d'un Opérateur envers une IA de Type Aide à la Décision
Trust in Automation : Analysing and Modelling Operator Trust in Decision Aid AI Over Time
Vincent Fer, Daniel Lafond, Gilles Coppin, Mathias Bollaert, Olivier Grisvard and Pierre De Loor

Collaboration Humain/Machine pour l'orchestration de drones: une plateforme d'expérimentation
Human-Machine Teaming For UAVs: An Experimentation Platform
Laila El Moujtahid, Sai Krishna Gottipati, Clodéric Mars and Matthew E. Taylor

Modélisation d'experts pour une assistance à base d'IA dans la lutte anti-sous-marine
AI-Driven Expert Modeling for Cognitive Assistance in Anti-Submarine Warfare
Tanya Paul, Daniel Lafond, Alexandre Marois, Yannick Alcaraz, Sébastien Deries and Gabriel Jourdon

IA & Apprentissage fédéré

Une implémentation GPU de la méthode de recherche approximative FlyHash
A GPU implementation of the approximate search method FlyHash
Arthur Da Cunha, Emanuele Natale, Damien Rivet and Aurora Rossi

Sécuriser la propriété intellectuelle dans l'apprentissage fédéré
Securing Intellectual Property in Federated Learning
Mohammed Lansari, Reda Bellafqira, Katarzyna Kapusta, Vincent Thouvenot, Olivier Bettan and Gouenou Coatrieux

IA & DRI

Détection et and identification radar de mini-drones à l'aide d'un réseau de neurone léger
Radar detection and recognition of micro-UAV using a lightweight neural network
Jean-François Degurse, Pierrick Richard and Ronan Guillamet

Développement de fonctions de coût pour l'entraînement de détecteurs radar robustes basés sur le deep learning
Loss Function Design For Training Robust Radar Detectors Using Deep Learning
Noé Lallouet, Tristan Cazenave, Cyrille Enderli and Stéphanie Gourdin

AI-based Text Generation for Semantic Search Robustness : Application to Defence

Claude Fendzi
Airbus Defence and Space
Elancourt , France
claude.fendzi@airbus.com

Bruno Carron
Airbus Defence and Space
Elancourt , France
bruno.carron@airbus.com

Guillaume Gadek
Airbus Defence and Space
Elancourt , France
guillaume.gadek@airbus.com

Abstract—**One of the biggest challenges in successfully applying Artificial Intelligence (AI) in the Defense sector is the availability of trustful domain specific data to train AI models on. These data have to be generated and collected from the real world or acquired through realistic scenarios simulations and validated by operation specialists or domain experts. In real world applications, most of the time these data are classified and difficult to access. Then only a handful coming either from unclassified documents or simulation / realistic scenarios can be made available. In this article, we discuss how Generative AI can be used to generate intelligence-oriented textual data that are semantically similar to a "ground truth" database. The methodology is applied in the frame of the EDIDP AI4DEF project, focusing on one of the use cases, Request for Information (RFI) semantic similarity detection in a database. We expose how a limited corpus has been enriched with noisy AI-generated data. The performances and the robustness of the AI model have been monitored to be kept similar before and after the data augmentation, while a human-in-the loop qualifies the AI-generated data.**

Keywords—Natural Language Processing, data generation, Large Language Models, Request for Information, Robustness, Semantic Similarity Search

I. INTRODUCTION

Natural Language Processing (NLP) fields have significantly evolved in the last couple of years, and gained a substantial paradigm shift with the advent of Large Language Models (LLM) [1], [2], [3]. These models, distinguished by their considerable size and comprehensive training data, have demonstrated extraordinary abilities in comprehending and producing human-like text [4], [5]. Recently, the transformer architecture was adopted by the GPT (Generative Pre-trained Transformer) series of models developed by OpenAI [5], including GPT-1 [6], GPT-2 [7] and GPT-3 [8]. These models were pre-trained on a large corpus of text from the internet, and then fine-tuned on specific tasks [9]. They demonstrated the ability to generate coherent and contextually relevant texts, marking a significant step forward in the field, beating the traditional NLP architectures (Recurrent Neural Networks, Long Short-Term Memory, Bidirectional Encoder Representation from Transformers), on various traditional tasks such as text processing or text understanding [10]. Traditional architectures were limited in their ability to model long-range dependencies in text, which is crucial for maintaining coherence over longer passages [11]. These new generations of transformer architectures achieved state-of-the-art results in NLP tasks on publicly available internet data, but to the best of our knowledge, have never been applied in the defence domain. Indeed, in the defence domain, and specifically in the intelligence domain, the data availability is a critical issue, due to confidentiality, security and classification. Thus, generating realistic intelligence domain-specific texts, relying on a generic model that was trained on open Internet data can be a tedious task, as the vocabulary and syntax in the intelligence domain are very specific, and may lead to the out-of-vocabulary (OOV) problem.

In this paper, we propose an approach to leverage large language models (LLMs) to generate intelligence domain specific textual data, keeping an *intelligence analyst* in the validation loop. These generated textual data considered as noise are then added in a reference intelligence text database. The approach is tested on a semantic similarity search task, and we compare the performance of our algorithms before and after the data augmentation. A focus is also made on the robustness to typos and misspelling in the search queries.

"RFI similarity search": comparing texts with respect to their semantics.

A very specific operational task in military headquarters consists in receiving RFIs (request for information)[1] that may bear on *any* topic relative to the current or future area of operations. RFIs are typically questions or notifications to ask either for information elements (whose complexity may range from the coordinates of a sensitive location, to the current security situation evolution perspectives between two ethnic groups), or to begin to be notified about topics. It is common to observe trends in the reception of RFIs, which are often bearing on similar topics or areas. Past RFIs, and their answers, are very likely to provide relevant answer elements; this is however not obvious to successfully retrieve these past RFIs without the dedicated support of the information system itself. We propose to perform this task with the help of an AI.

[1] https://www.intelligence101.com/an-introduction-to-the-intelligence-cycle/

In this paper, we propose a candidate approach to tackle the problem of lack of data in RFI, especially in the defence domain. We bootstrap from a limited number of reference sample data to generate new ones that are semantically similar. In details, our contributions are the following:

- we introduce a specific challenge of NLP: RFI semantic similarity detection.
- we propose and implement a methodology to tackle the lack of available real annotated data in the text modality.
- we propose a method to evaluate the generated data quality, benefiting from the expertise of qualified *human operators.*
- we assess the robustness of the data augmentation process to the RFI semantic similarity detection task.

The rest of the paper is structured as follows: Section II formalizes the operational problem of RFI semantic similarity search and the methodology that has been implemented. Section III describes the datasets and the experimental setups. The results of these experiments are discussed in sections IV and V; Section VI concludes this work.

II. Semantic Similarity Search For Defence

A. RFI Similarity Search: Operational Requirements and Hypotheses

The RFI similarity detection activity describes a task performed by an *intelligence analyst*, who receives a Request for Information (RFI) from a *Commander* and tries to find if there exists a similar RFI within a Database (typically, the Intelligence Collection Plan database). From a purely technical point of view, the system relies on an AI algorithm that implements a semantic similarity search engine that allows to compute the RFI similarity in a RFI database. The algorithm supports the *intelligence analyst* in their task: upon creation, the new RFI will automatically be associated with suggestions of past RFIs, along with their answers.

- **1st step:** considering a "first" RFI semantic search query, the system is expected to highlight similar RFIs existing in the database along with a confidence level.
- **2nd step:** the analyst then takes advantage of the retrieved RFIs to better improve their RFI request. This helps them to create more accurate RFIs and / or to collect RFI products which best match the original RFI from the *Commander*. This leads to both time saving and accuracy gain in search results.

Operational constraints and requirements- of the Intelligence Collection Plan Database

The *intelligence analyst* receives the *Commander's* needs and has to write the RFI. The main operational requirements are:

- the needs expressed by the *authority (Commander)* shall be properly taken into account in the RFI request form,
- the RFI query shall fill at least mandatory fields (RFI subject or title, RFI Details, RFI Date information, Areas or geographical zones),
- the system shall not duplicate an existing RFI,
- the system shall create the RFI in the shortest possible time.

Hypotheses

We consider the following hypotheses to support the experiment:

- as a prerequisite, there exists a RFI database (*ground truth*) that supports the search query and gets semantically similar RFI if any.
- the RFI database contains all the information enabling the search,
- the RFI database is realistic and semantically correct to lead to accurate and robust search results,
- the RFI database has been validated by an operation specialist *(intelligence analyst*)

B. Methodology to assess the robustness to noise

This study examines how an intelligence domain specific AI-based text-generator can be used to augment or enrich a *ground truth* database which supports the RFI semantic similarity search. The experiment consists in semantically indexing the "manually" validated database in the training step, and performing a semantic search query to retrieve semantically similar RFIs in the inference step. This process is first executed on the ground truth RFI database. Then, the database is augmented with an AI-based text generator and the experiment is repeated. The purpose of this second step is to examine and assess the robustness of the AI-based RFI semantic search algorithm to the noisy generated data. Assessing robustness to noise is necessary in all NLP tasks, as most of the task-specific datasets are not fully representative of the infectivity of the potential user inputs. In the literature, datasets of reference can be perturbed (adding spelling errors, casing, modifying the order of the words or using translation back and forth) either to train the model on noisier data or to evaluate its robustness to noisy inputs [17]. In particular we expose how a limited corpus has been enriched with noisy AI-generated data. The performance and the robustness of the AI model are monitored to be kept similar before and after the data augmentation, while a human-in-the loop qualifies the AI-generated data. A special focus is made on the typos, synonyms and spelling errors in the RFI query.

C. Implementation architecture of the semantic similarity search on RFIs

The implementation of the semantic search algorithm consists of 2 steps. The first step is the training phase from historical RFI data (vector space representation of the RFI database), and the second step concerns the inference phase on new RFIs. The training phase consists in building machine learning models that encode the RFI database elements into a vector space representation while preserving the semantics of the corpus. The trained models are then saved and ready to be used in the inference phase. Fig.1 illustrates the 2 phases.

The training phase builds the word embeddings of the corpus associated with the RFI database. This embedding represents an encoding result which is a low-level representation of the RFI corpus into a vector space. Four different algorithms have been implemented for this study. A deep learning-based architecture derived from the encoder transformer's

architecture (Universal Sentence Encoder) [12], a latent semantic indexing algorithm (LSI) [13], a TF-IDF (term-frequency - inverse document frequency) algorithm [14] and finally a fuzzy string match (FSM) algorithm [15].

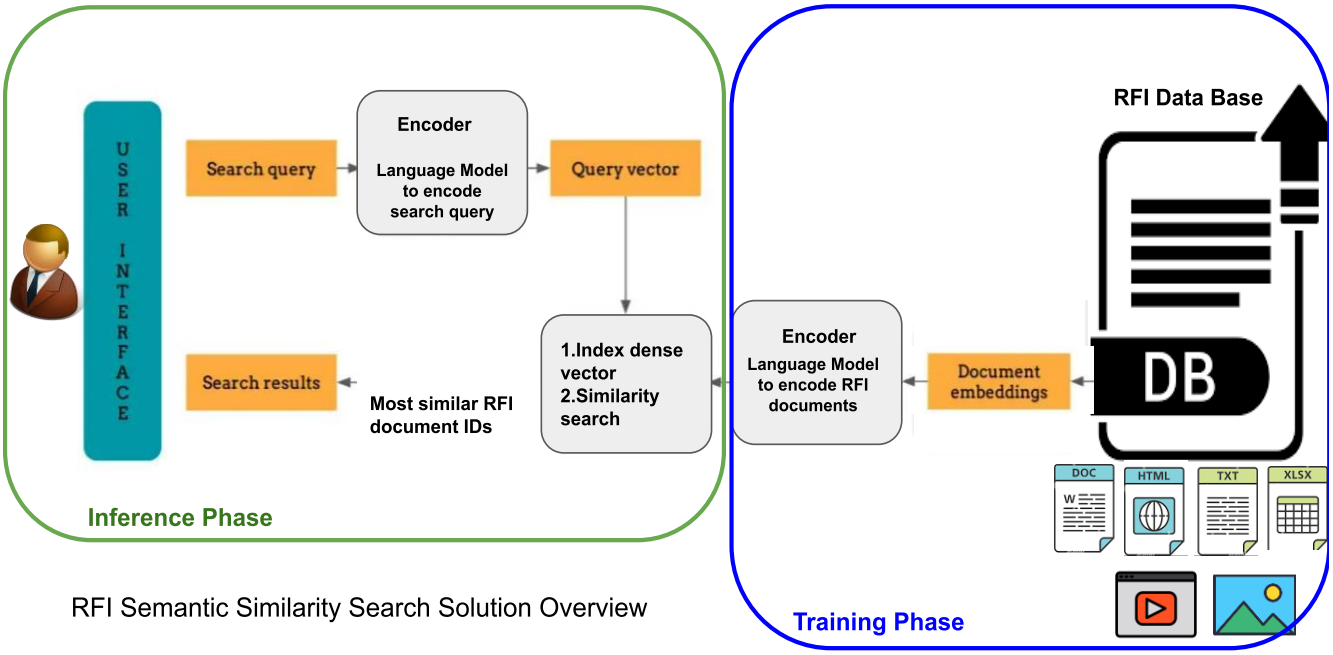


Figure 1: RFI Semantic Similarity Search Solution Overview

For the deep-learning architecture, the context-based encoding method used is the Universal Sentence Encoder, which is a model based on a transformer architecture that considers both the word order and the identification of the remaining word in the sentence. The LSI instead analyses a set of documents in order to discover statistically relevant co-occurrences of words or terms. It uses a document-term matrix decomposition of the corpus of document to cast queries into a low-rank representation vector space, enabling to compute query-document similarity scores in this low-rank representation vector space. The matrix decomposition can be updated with new observations at any time, for an online, incremental, memory-efficient training.

III. Datasets and Experiments Setup

A. *Datasets*

Handcrafted datasets

A database of 300+ entries of manually created RFIs in the STANAG[2] format have been created for this study. This RFI database has been built by an *intelligence analyst* to better represent the RFI data structure and mandatory fields. The RFI database contains the following fields:

- a synthetic *subject* description or RFI *title*
- *details* if any, corresponding to additional details enriching the RFI *title* (free text).
- areas of interest concerning the needs, describing the *area* or *zone* of interest to focus the search in.
- the authority identifier (*Commander*)
- the recipients in action and in information
- the date parameters "Not Earlier Than" & "Not Later Than"

In the scope of this study, only textual fields have been considered as mandatory: *Subject* and *RFI Details*. The Table I below shows a caption of a realistic RFI sample database.

TABLE I: Example of RFI sample database with FRI *Subject* and *Details* fields. The *intelligence analyst* tries to find semantic similarity between the *commander's* request and the RFI database (*Subject* or *Details*)

Subject	Details
Anti-aircraft threats on Coalition Forces within the operation area XXX	Enemy forces have anti-aircraft means deployed in XXX. It has short- and medium-range anti-aircraft systems dating from the 1980's, including the YYY and ZZZ ground-air defence system. It includes relatively low-tech anti-aircraft guns, to deal with the jets, helicopters, drones, and missiles. What are the threats represented by these assets?
Iraqi theatre - Factors of unrest in the civilian population in the North region	What are the potential factors of unrest in the civilian population in northern Iraq?

[2] https://nso.nato.int/nso/nsdd/main/list-promulg

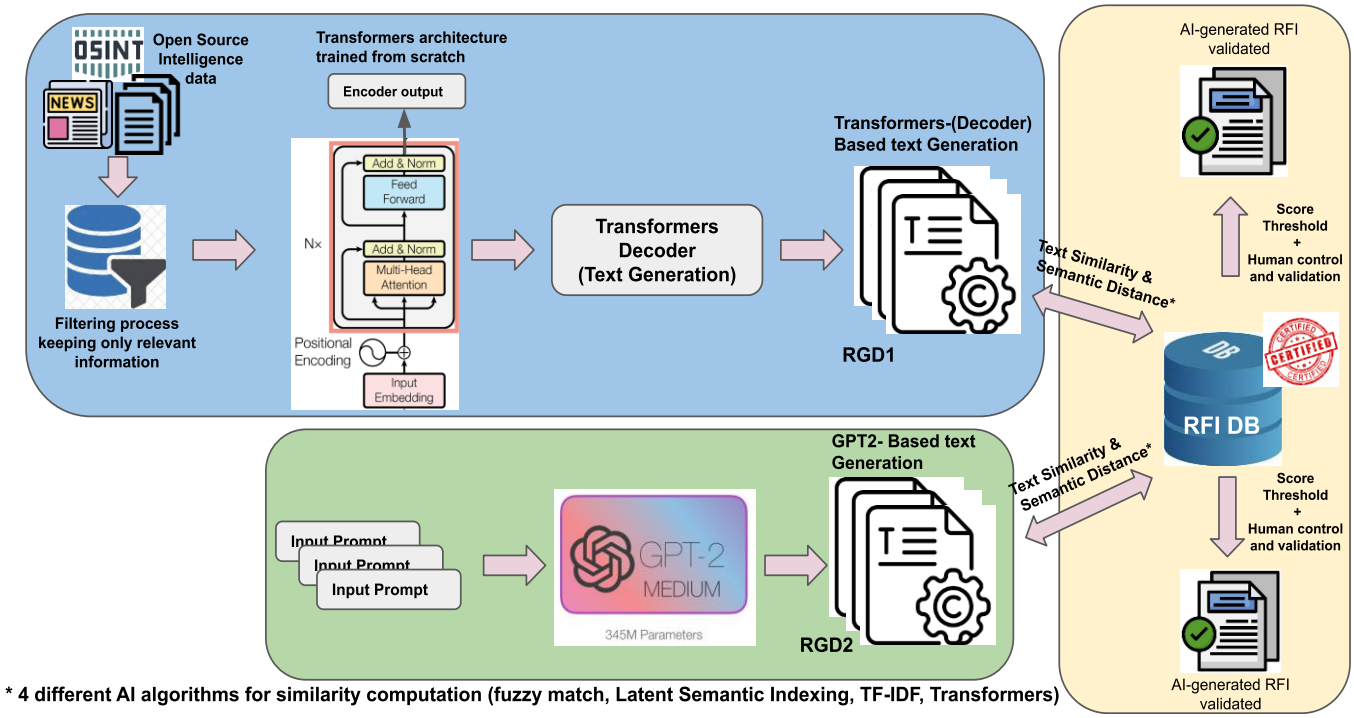


Figure 2: AI-based Intelligence Domain-specific Text Generation and Validation

AI-based text generation for augmenting the reference RFI database

The AI-based intelligence domain specific text generation and validation is described in Figure 2. Two datasets have been generated. The first one (RGD1) has been generated after training from scratch an encoder-decoder architecture (keras_nlp [3] transformer encoder, 12 heads, embedded dimension 64) on open-source intelligence data [456789], including release press papers, domain-specific intelligence journals, intelligence reports, etc. (blue box in Fig. 2). The second dataset (RGD2) has been generated using the GPT2 pre-trained model (green box in Fig. 2). We used the *GPT2-medium*[10] model with 345M parameters. We extracted the *title* of all the RFIs in the *ground truth* database (manually generated RFI datasets) and used them as a prompt for text-generation. For both datasets, the tokens generated have been monitored to be semantically "close" to the RFI *ground truth* database.

For each set of generated tokens, a semantic similarity score has been computed and these tokens are discarded if the score (cosine distance) is less than a predefined threshold in addition to the *human control and validation* (mustard yellow box in Fig. 2). The predefined threshold value will be discussed in the next section. The AI-based text-generated data are then used to augment or enrich the *ground truth* RFI database. From the process described in Fig. 2, we have generated 17,820 additional RFI.

B. Experiment setup

In this section, the setup experiments carried out for both the realistic manual generated RFI dataset and the AI-generated RFI data set is discussed. The RFI semantic similarity detection is performed on both datasets and the performances are monitored to be kept similar.

1) Experiment 1: evaluation on handcrafted data

The training and inference phases for RFI similarity detection described in section III.D are implemented on the manually generated RFI datasets (ROVEY dataset: RFI Operator Validated EntrY). All of the four algorithms are considered here and a set of semantic search similarity metrics are computed. The common similarity metric used in the semantic search to measure the similarity/distance between vectors have been considered for this study, which is cosine similarity. In addition to this metric, we have also considered the *recall*, measuring the ability of a search engine to find the relevant material in the index, and the *precision*, measuring its ability to place that relevant material high in the ranking. But these latter metrics have only been computed for the first experiment executed on the handcrafted RFI dataset, as the intelligence analyst helped in performing the time-consuming annotation task on this dataset.

[3] https://keras.io/guides/keras_nlp/transformer_pretraining/
[4] https://www.foreignaffairs.com/browse/snapshot
[5] https://intpolicydigest.org/domestic-extremist-groups-pose-a-unique-challenge/
[6] https://intpolicydigest.org/essential-breakthrough-in-kazakhstan-uzbekistan-relations/
[7] https://documents.theblackvault.com/documents/dia/Afghanistan_Stalemate_Continues_CLEAR.pdf
[8] https://nsarchive.gwu.edu/document/29551-42-annual-threat-assessment-switzerland
[9] https://nsarchive.gwu.edu/document/18362-national-security-archive-estonian-foreign
[10] https://huggingface.co/gpt2-medium

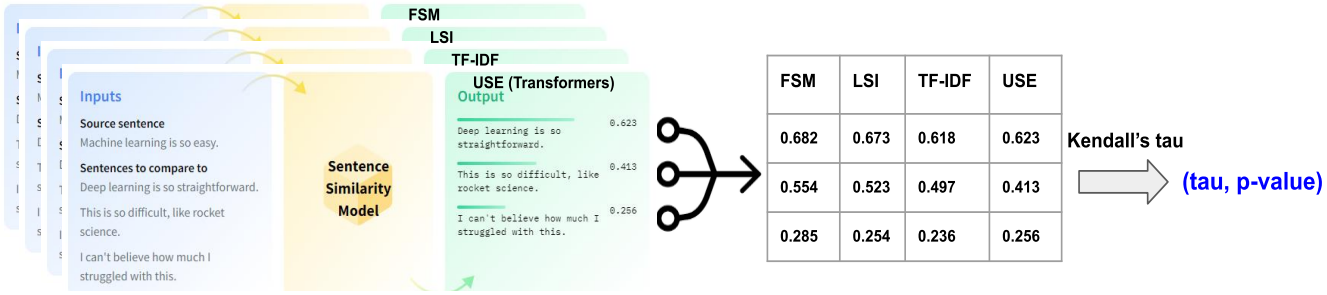


Figure 3: Robustness evaluation process

2) Experiment 2: evaluation on synthetic data

The second experiment implements the same RFI semantic similarity search algorithm on an AI-generated dataset (RFI GPT Data: RGD1&2). The training and inference processes described in experiment 1 are thereafter performed. For Experiment 2, the three similarities metrics described in the previous section below are also computed and they are compared with the ones obtained in Experiment 1.

For each of these two experiments, the robustness of the system is assessed using Kendall's tau correlation coefficient [16] on the similarity search scores for all of the 4 search algorithms (FSM, LSI, TF-IDF, USE). Fig. 3 illustrates the evaluation process. A total of 168 individual queries have been performed, and for each single query and each single algorithm, we retrieve the top-k more similar items in terms of score and we compute the Kendall's tau correlation coefficient between the score tables from one algorithm and another one (e.g., LSI vs TF-IDF). Kendall's tau is a non-parametric measure of relationships between columns of ranked data. The Kendall's tau correlation coefficient returns a value between 0 and 1, where 0 shows no relationship and 1 is a perfect relationship. Kendall's tau has been chosen because it has good statistical properties and its interpretation in terms of the probabilities of observing concordant and discordant pairs is very direct. Thus, if the Kendall's tau is in the same range, before and after the data augmentation, then the semantic search algorithms are robust to data augmentation.

IV. Results

A. Accuracy

The accuracy of the search algorithms has been assessed first in terms of the relevance of the RFI responses. For each RFI query, the search results are carefully analyzed by the *intelligence analyst*. They then attribute a binary rating (0: Bad response, 1: Good response) to the overall search results in a very conservative way, and provides some free-text comments about the accuracy, the relevance and the robustness of these results. Each query is considered as one trial. The similarity score ranges from 0 (no similarity) to 1 (perfect match). 45 individual trials (query) have been performed and a naive statistical count of the good responses versus the bad ones have been done. We obtained 36 good responses and 9 bad responses which gives a global "accuracy" score of 80% with a database of 300 RFI entries (ROVEY dataset).

B. Robustness

A set of 168 individual queries have been performed for each of the four similarity algorithms; only the top-20 similar RFIs have been selected for each query and the threshold value of the similarity score (cosine similarity) has been fixed to 0.35. If a result has a score inferior to this threshold value, we do not consider it (i.e., no relevant match). In addition, we only consider query results with more than five RFI responses to have "enough" sample data to compute Kendall's tau coefficient. In order for a response to be considered, it has to be captured by each of the four algorithms. Table II represents an illustration of the ranked similarity score obtained for one query for each of the four algorithms. In this example, the query's response showed 11 similar RFIs with similarity scores greater than 0.35. The Kendall's tau coefficient and the associated p-values are computed considering pairs of data in Table II. The results are shown in Table III.

TABLE II: Ranked similarity scores for all of the four algorithms (single query)

	Fuzzy	LSI	TF-IDF	USE
0	1	1	0.89	1
1	0.92	0.92	0.82	0.95
2	0.86	0.92	0.82	0.92
3	0.81	0.92	0.73	0.87
4	0.79	0.92	0.73	0.85
5	0.77	0.85	0.73	0.82
6	0.74	0.84	0.56	0.77
7	0.63	0.84	0.49	0.71
8	0.61	0.84	0.49	0.64
9	0.61	0.84	0.46	0.58
10	0.6	0.78	0.43	0.57

TABLE III: Kendall's Tau coefficient and the associated p-values for score data in Table II (to read (Tau, p-value))

	Fuzzy	LSI	TF-IDF	USE
Fuzzy	(1.0, 2e-05)	(0.892, 0.00039)	(0.943, 0.0001)	(0.991, 2e-05)
LSI		(1.0, 0.00019)	(0.884, 0.00061)	(0.884, 0.0004)
TF-IDF			(1.0, 6e-05)	(0.953, 7e-05)
USE				(1.0, 0.0)

Table III shows on average a quite high Kendall's tau correlation coefficient (with minimum value greater than 0.88), meaning a strong correlation between all of these algorithms. Moreover, the p-values are low (close to 0), thus

rejecting the null hypothesis (that there would be an absence of association between two algorithms). Kendall's non-parametric statistical test has been considered to succeed if Kendall's tau value is greater than 0.8 and the p-value less than 0.03.

For the original RFI database (ROVEY dataset), among the 168 individual queries performed for each algorithm, 84 retrieved at least 5 similar RFI with a similarity score greater than 0.35. Moreover, these figures were the same for all of the four algorithms, leading to a 50% "good" RFI query matching. Among these 84 queries, all of them passed the Kendall test, which is Kendall's tau coefficient is greater than 0.8 and its p-value less than 0.03 as stated above. The same experiment has been performed after the RFI data generation and augmentation. From the process described in Fig. 2, we have generated 17820 additional RFI (RGD data), considered as noise in the original ROVEY dataset, as the upper bound of the similarity score with the later DB was kept quite low at 0.2 in order to get more RFI entries to enrich the original DB. Summary of these results are presented in Table IV, where we exhibit results with and without data augmentation. As the RFI generated data are considered as noise in the original RFI database, the figures in Table III show that the different performance indicators do not evolve at all before and after the data augmentation, which makes all of the similarity search models robust to noisy data augmentation. These stable results are first due to the noisy nature of the generated data, and secondly, due to the upper bound value of the similarity score for RFI data generation (0.2) which is less than the threshold score value considered for all of the query's responses (0.35).

TABLE IV: Overall Key Performance Indicators (KPI) with and without data augmentation

Parameters to be monitored	**RFI DB without Data Augmentation**	**RFI DB with Data Augmentation**
Total Nb or queries	168	168
Nb of queries that crossed the score threshold[11]	84	85
Nb of queries that passed the Kendall's statistical test	84	85
Nb of queries that failed the Kendall's statistical test	0	0

Another figures to look at are how the system behaves with and without the data augmentation. To do so, among all the queries that passed the score threshold and Kendall's statistical test, we identified the intersection between the 84 and 85 responses queries. This intersection was identified based on the similarity in the query's input text for the search and the equivalence in the output results. From this subset of similar queries responses between RFI DB with and without data augmentation, we computed for each individual query, how the number of retrieved RFI document behaved. In other words, we check whether the number of retrieved RFIs changes before and after data augmentation, taking as reference this number before data augmentation. We also focus on whether the order (ranked by similarity score) of the responses is preserved before and after the data augmentation. Table V shows that a high percentage of RFI responses were observed both before and after the data augmentation (85.7%), among them 66.7% preserved the number of RFI retrieved, while 30.5% led to the same number of RFI responses, before and after the data augmentation. The order in terms of ranking of the similarity score was preserved in 23.6% of cases.

TABLE V: KPI at individual query's response level, taking as reference the RFI database without data augmentation

KPI - Intersection	**KPI - Inclusion**	**KPI - Preserve same Nb of RFI responses**	**KPI - Preserve same order in RFI responses**
72	48	22	17
85.7%	66.7%	30.5%	23.6%

KPI - Intersection: among the (84, 85) pairs of queries that passed all the tests (threshold score and Kendall's test), how many individual response pairs passed the test?

KPI - Inclusion: among the Intersection, how many preserved the number of RFI responses compared to the results without data augmentation?

KPI - Preserve same Nb of RFI responses: among Intersection, how many preserved the same number of RFI responses compared to the results without data augmentation?

KPI - Preserve same order in RFI responses: among the Intersection, how many preserved the same order (ranked similarity score) in RFI responses compared to the results without data augmentation?

V. Discussions

The two experiments performed in this study showed quite similar results both qualitatively and quantitatively for the RFI semantic search task. The overall model accuracy was estimated at 80% for the semantic search task on ROVEY dataset, while the robustness to noise was assessed using various techniques (Kendall's tau and statistical analysis of queries output). The semantic similarity search score threshold was fixed at 0.35 to consider only relevant RFI responses to compute the performance metrics, and the RFI response's selection process was rather conservative both in terms of Kendall's tau value and its associated p-value. The metrics displayed in Tables III, IV and V show an overall good performance of our approach, making it a good candidate to tackle the lack of data to train AI models on in the defence sector.

[11] Similarity scores greater than 0.35 for each query's response and each algorithm and a minimum of 5 similar RFI retrieved for each algorithm

VI. Conclusion and future work

The study examined how an intelligence domain specific AI-based text-generator can be used to augment or enrich a *ground truth* database which supports the RFI semantic similarity search. In particular we exposed how a limited RFI corpus has been enriched with noisy AI-generated data, and assessed the performance and the robustness of the AI model to this noisy data when performing RFI semantic similarity search. We assessed the performances of the search engine and showed that they were kept similar before and after the data augmentation, making the search algorithms robust to data augmentation in particular noisy data. The assessment relied on a *human-in-the loop* (*intelligence analyst*) who annotated and qualified both the handcrafted data and the AI-generated data. The different results discussed in this study make the proposed approach a good candidate to deal with the lack of data when designing and training AI systems in the defence sector. It is worth mentioning that the approach was tested on handcrafted realistic RFI data, and needs to be validated on real data from operations. Moreover, the annotation step and model accuracy assessment would have to be performed by different *end-users* in order to reduce as much as possible the underlying bias. These open questions have to be addressed in future studies.

Acknowledgment

This project has received funding from the European Defence Industrial Development Programme (EDIDP) under grant agreement No: EDIDP-AI-2020-066-AI4DEF.

References

[1] Sébastien Bubeck, «Sparks of artificial general intelligence: Early experiments with gpt-4» *arXiv* , 2023.

[2] Aakanksha Chowdhery, «Palm: Scaling language modelling with Pathways» *arXiv*, 2022.

[3] Hugo Touvron et al, «LLaMA: Open and Efficient Foundation Language Models» *arXiv*, 2023.

[4] OpenAI, «Introducing chatgpt,» 2022. [En ligne]. Available: https://openai.com/blog/chatgpt.

[5] OpenAI, «GPT-4 Technical Report» *arXiv* , 2023.

[6] A. Radford et al. «Improving Language Understanding by Generative Pre-Training» 2018.

[7] A. Radford et al., «Language Models are Unsupervised Multitask Learners,» 2018.

[8] T.B. Brown et al., « Language models are few-shot learners» *arXiv* , 2020.

[9] A. Vaswani et al. «Attention is all you need» *Advances in Neural information processing systems*, vol. 30, n°130, 2017.

[10] Haifeng Wang et al. « Pre-Trained Language Models and Their Applications» *Engineering*, n° %1 ISSN 2095-8099, 2022.

[11] J. Hochreiter et al. «Long short-term memory,» *Neural computation*, vol. 8, n°19, pp. 1735-1780, 1997.

[12] Daniel Cer et al. , «Universal Sentence Encoder,» *arXiv*, 2018.

[13] S. Deerwester et al., «Indexing by latent semantic analysis» *Journal of the American Society for Information Science*, n°141, 1990.

[14] G. Salton et al., «Term-weighting approaches in automatic text retrieval» *In Information Processing & Management*, vol. 24, n°15, pp. 513-523, 1988.

[15] Rares Vernica et al., «Efficient top-k algorithms for fuzzy search in string collections» *Proceedings of the First International Workshop on Keyword Search on Structured Data*, pp. 9-14, June 2009.

[16] L. Laurencelle, « Le tau et le tau-b de Kendall pour la corrélation» *Tutorials in Quantitative Methods for Psychology*, vol. 5, n°12, pp. 51-58, 2009.

[17] J. Náplava, et al. «Understanding model robustness to user-generated noisy texts» *arXiv* , 2021.

Towards simulation of radio-frequency component with physics informed neural networks

Mathieu Riou
SINCLAIR Lab
Thales Research and Technology
Palaiseau, France
mathieu.riou@thalesgroup.com

Mouhamed Coulibaly
SINCLAIR Lab
Thales Land and Air Systems
Limours, France
mouhamed.coulibaly@ensta-paris.fr

Jean-Pierre Marcy
Thales Land and Air Systems
Limours, France
jean-pierre.marcy@thalesgroup.com

***Abstract*—Radio-frequency components such as transmission lines are present in many defense systems such as radars, communication systems or electronic warfare systems. The conceptions of radio-frequency components involves a simulation phase that generally consists in the resolution of harmonic Maxwell's equations. Presently, these simulations are performed with mesh based methods such as finite element or finite difference methods. The results of a simulation is specific to a given geometry and mesh and can not be reemployed for a similar problem. The recently introduced physics informed neural networks (PINNs) could offer a mesh-free alternative to these methods. While PINNs for fluid mechanics has been tested multiple times, there are only few examples of PINNs for harmonic Maxwell equations in two dimension. In this paper we present a first attempt of training a PINN for the three dimensions simulation of a transmission line and we highlight the specific difficulties of training a PINN for radio-frequency simulations.**

***Index Terms*—deep learning, physics, physics informed neural network, radio frequency, electromagnetism**

I. Introduction

Radio-frequency components such as transmission lines are present in many defense systems such as radars, communication systems or electronic warfare systems. The design process of these components integrates a simulation phase. Simulations are performed with mesh based methods such as finite element or finite difference methods that involve a discretization of the space. In particular the computation of transmission property of these component may involve to compute the electric field for various boundary conditions. This computation requires to solve the harmonic Maxwell's equation. Such computation may be slow especially for higher frequency where the characteristic size of the mesh elements should be decreased. Moreover a complete characterization of component requires multiple simulations for instance varying the frequency of the field for analysing the frequency-dependent transmission and the shape of the component for sensibility analysis.

Physics informed neural network (PINN) [13] are an alternative to mesh based method to speed up simulations of filter. Firstly such methods are mesh free, and thus the tedious task of defining a mesh is avoided. Secondly they allow for leveraging all advantages of neural networks such as transfer learning which speed up the computation for similar problems, for instance similar geometries or similar frequencies. PINNs consists in integrating physical knowledge during the training step of the neural network. Such methods have been deployed for direct and inverse problem [13], [2]. PINN have been tested in multiple physics mostly in fluid mechanics [1], but also in solid mechanics, thermodynamics, chemistry and nano-optics [11], [3]. However, the use of PINN in electromagnetic is still at its infancy with examples for meta-material design [7],[10]. However these two examples present simple geometry only in two dimensions such as square domain and problems with Dirichlet boundary conditions where the field value is known in every boundaries. The design of radio frequency component for real world systems requires to model complex three dimensional geometry with non-Dirichlet boundary conditions. This study tests the implementation of PINN for the simulation of a resonant transmission line with a stub made in tri-plate air technology. Such transmission lines are used for high power radio-frequency circuits such as the one in long-range air defence radars. The simulation is in three dimension with non-Dirichlet boundary conditions and allows for challenging the PINN. The contributions of this work are:

- The implementation of a PINN for solving the harmonic Maxwell's equations on a transmission line.
- The highlight of the specific difficulties linked to training a PINN on non-Dirichlet Boundary conditions
- The test of an adaptive training strategy for balancing the different boundary conditions.

II. Principle of physics informed neural network

Also there were preliminary works [6], the seminal papers for the PINN were published by Raissi *et al* first in two paper in 2017 and than in 2019 [13]. The approach attracted much interest due its genericity (it can adapt to any problem described by a partial differential equation) and its relatively easy implementation as it leverages standards machine learning libraries to perform autoderivation for computing the residual that is at center of the approach. Considering a problem governed by an partial differential equation (PDE):

$$\mathcal{N}_x(u) = 0 \quad in \quad \Omega$$

Where $\mathcal{N}_x$ is a diferential operator and u is the solution and Ω is the domain where the solution is searched. The problem is also constrained by a set of boundary conditions:

$$f_b^i(u) = 0 \quad on \quad \partial\Omega_i \quad for \quad 1 \leq i \leq n_b$$

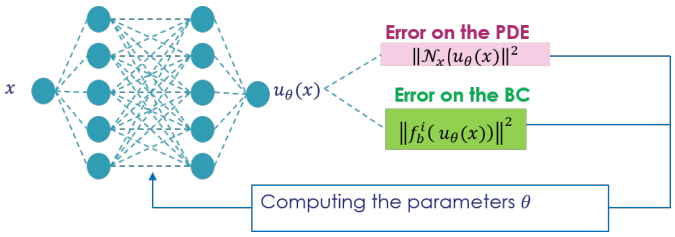


Fig. 1. Principle of a PINN

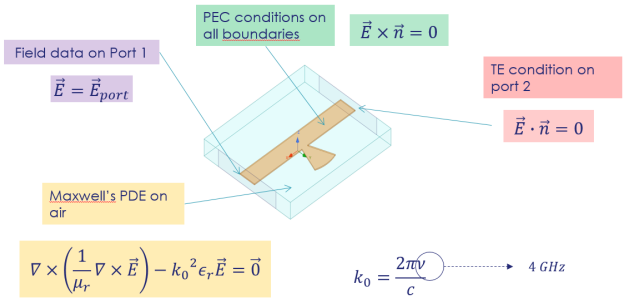


Fig. 2. Physics of the problem

$\partial\Omega_i$ is a boundary of Ω and n_b is the number of different boundary conditions. The idea behind PINN is to use a neural network $u_\theta(x)$ to approximate the solution of the problem $u(x)$. The motivation for using a neural network as solution basis comes from the universal approximator property of the feedforward neural networks [9], [4]. A neural network with a unic hidden layer and finite number of neurons can approximate arbitrary closely any continuous function. The neural networks parameters θ should than be trained in order to approximate the solution. The second characteristic of PINN is to embed the physics bias in the cost function used to learn the parameters of the network. The Figure 2 illustrates the working principle of a PINN.

The parameters θ of the neural network are optimized such that the error terms on the PDE, on the boundary conditions (BC) and on potential known data are minimized. The architecture of PINN are generally feed forward neural networks, and the automatic derivation methods are used to compute the loss term associated to the PDE. The computation of the losses involve also a sampling step, where points in the domain for the evaluation of the PDE error and boundary condition error are chosen. PINNs with only the PDE loss and the boundary conditions loss can solve direct problems [13]. With the addition of observation data but with unknowns in the PDE, PINNs can solve inverse problems or retrieve hidden physics [2], [14].

III. Usecase definition

This work study a relatively simple transmission line with a stub. This stub gives rise to a resonant behavior. The problem formulation is illustrated in Figure 2. The electromagnetic field should respect a PDE (the harmonic Maxwell's equations) and 3 boundary conditions on different domains detailed later. For this problem, the domain can be thus divided in 4 areas. For each domain one of the four physical constraints must be respected. In Figure 2 for each area of the transmission line, we represent the condition respected by the electrical field. Simulations will be carried out for a frequency of 4GHz.
In the airbox the electric field $\vec{E}$ has to verify the Maxwell equation in harmonic regime (highlighted in yellow in Figure 2):

$$\boldsymbol{\nabla} \times \left(\frac{1}{\mu_r}\boldsymbol{\nabla} \times \vec{E}\right) - k_0^2 \epsilon_r \vec{E} = \vec{0}$$

Were $\vec{E}$ is the electric field, μ_r is the relative permeability of the medium, k_0 is the wave number in vacuum, ϵ_r is the relative permittivity of the medium. In our case we consider that the electric field is in vacuum so $\mu_r = 1$ and $\epsilon_r = 1$. The dependency to the frequency of the signal is embedded in the wave number k_0:

$$k_0 = \frac{2\pi\nu}{c}$$

With ν the field frequency and c the speed of light. On both boundaries of the airbox and on transmission lines the electric field has to respect the perfect electric conductor (PEC) condition which implies that the transverse terms of the electric field should be equal to zero at the surface:

$$\vec{E} \times \vec{n} = \vec{0}$$

On the exit port, transverse electric field (TE) conditions is imposed, which means that the normal component of the electrical field is null.

$$\vec{E}.\vec{n} = 0$$

The entry port the electric field is known:

$$\vec{E} = \vec{E}_{port}$$

IV. PINN with a non-adaptive training

A. Neural network architecture

The physics informed neural network should take as input the variables of the field and should return as an output the field of interest. In our case the field is the electrical field $\vec{E}$ which has three components (E_x, E_y, E_z). $\vec{E}$ is a function of the coordinates (x, y, z). The input of the neural network is thus the coordinates (x, y, z) and the output is a prediction of $\vec{E}$ that are referred as $\hat{E} = (\hat{E}_x, \hat{E}_y, \hat{E}_z)$. Relying on what have been done in the literature a fully Connected neural network is used to train our model (Figure 3). For convenience only four layers of neurons are shown in Figure 3 including an input layer for the coordinates (x, y, z), an output layer for the prediction $(\hat{E}_x, \hat{E}_y, \hat{E}_z)$ and two hidden layers. The neurons for hidden layers are differentiable non-linear functions called activation functions. The choice of a neural network architecture consist in choosing the number of layers, the type of layers (dense or for example convolution etc...), the number of neurons per layer and the type of activation function

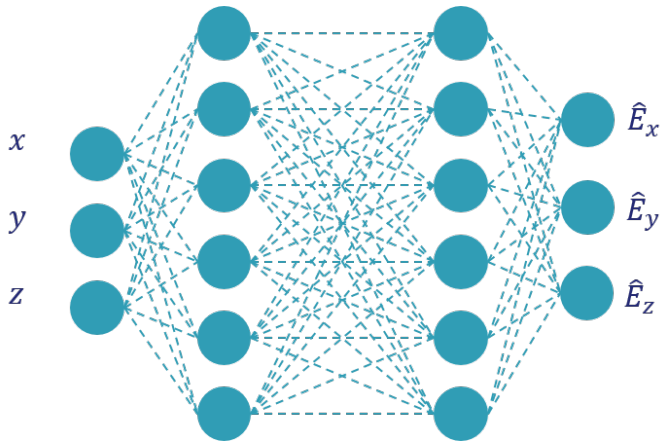


Fig. 3. Fully connected neural network architecture for the prediction of electrical field

(for example hyperbolic tangent). For the rest of our study, a neural network 10 layers of 50 neurons with hyperbolic tangent activation function will be used. The fact the neural network takes $\vec{x}$ as input allows for computing $\nabla \times \nabla \times \hat{E}$ by derivating twice the output of the neural network with regards with its input. This derivation is performed with the automatic differentiation libraries that are also used for the training of neural networks.

B. Loss terms

To adapt PINN method in our context the different terms of physics knowledge should be integrated during the learning process. Information on physics will be taken into account by computing the residual on each physics constraint (see Figure 2). Four different losses enforce the physical conditions described in the previous section on the four different model *ie* respectively the Maxwell's differential equation in the airbox (higlighted in orange), the source term on the entry port (highlighted in purple), the perfect electric conductor condition highlighted in green and the transverse electric field condition highlighted in red. $\mathcal{L}_{PDE}$ is the loss measuring the error of the neural network with regard to the Maxwell's equation in the airbox. This loss correspond to the norm of the residual that the neural network makes on Maxwell's equation.

$$\mathcal{L}_{PDE} = \frac{1}{n}\sum_{i=1}^{n} \|\boldsymbol{\nabla} \times \left(\frac{1}{\mu_r}\boldsymbol{\nabla} \times \hat{E}(\overrightarrow{x_i})\right) - k_0^2 \epsilon_r \hat{E}(\overrightarrow{x_i})\|^2$$

Here $\hat{E}$ is the electric field predicted by the neural networks. $\overrightarrow{x_i}$ design coordinates vectors of points sampled inside the airbox. n is the number of points chosen in the airbox to evaluate this loss.

$\mathcal{L}_{source}$ is the loss measuring the respect of the source term at the entry port (port 1). This loss is simply the quadratic error between the prediction of the neural network and the real electric field.

$$\mathcal{L}_{source} = \frac{1}{m}\sum_{k=1}^{m} \|\hat{E}(\overrightarrow{x_l}) - \vec{E}(\overrightarrow{x_l})\|^2$$

$\vec{E}$ is the real electric field. $\overrightarrow{x_l}$ are coordinate vectors of point on the port 1. m is the number of point chosen in on the port 1.

$\mathcal{L}_{PEC}$ is the loss ensuring that the perfect electrical condition is enforced on the metallic boundaries (boundaries of the airbox excepted to the port and boundaries of the transmission lines).

$$\mathcal{L}_{PEC} = \frac{1}{p}\sum_{k=1}^{p} \|\hat{E}(x_k) \times \vec{n}(\vec{x_k})\|^2$$

The loss is computed as the norm of the vector product between the electric field predicted by the neural network $\hat{E}$ and the normal vector $\vec{n}$. This norm is evaluated on m points whith coordinates $\vec{x_l}$ belonging to the metallic boundaries domain.

Finally, $\mathcal{L}_{TE}$ is the loss ensuring that the transverse electrical field condition is respected by the predicted field.

$$\mathcal{L}_{TE} = \frac{1}{q}\sum_{h=1}^{q} \|\hat{E}(x_h).\vec{n}(x_h)\|^2$$

It is expressed as the norm of the scalar product between the predicted field $\hat{E}$ and the normal vector $\vec{n}$ and is evaluated on q points belonging to the exit port (port 2) of coordinate $\vec{x_h}$. At this point it is worth noticing that the knowledge of the filter geometry is crucial to sample the point $\vec{x_i}$, $\vec{x_l}$, $\vec{x_k}$ and $\vec{x_h}$. The constraints of the geometry is embedded in the different loss terms because of their dependency to the choice of the points $\vec{x_i}$, $\vec{x_l}$, $\vec{x_k}$ and $\vec{x_h}$ also referred as collocation points. In order to embed the physics of the problem, the parameters should be trained such that the four different loss terms $\mathcal{L}_{PDE}$, $\mathcal{L}_{source}$, $\mathcal{L}_{PEC}$ and $\mathcal{L}_{TE}$ are minimized.

The most naive way to aggregate the different loss terms in a single loss function is to sum them:

$$\mathcal{L} = \mathcal{L}_{PDE} + \mathcal{L}_{PEC} + \mathcal{L}_{TE} + \mathcal{L}_{source}$$

The following section will present the results of the training choosing this loss.

C. Training and results

The training is performed on 30 000 epochs for the fully connected neural network with only $tanh$ activation functions. The batch size for each error terms is 500 for about 40 000 collocation points. In order to select the collocation points, the geometry is first divided in canonical geometrical entities (here portion of cylinders and rectangular parallelepiped). Point are then sampled randomly in these canonical elements using random sampling in euclidean coordinates (x, y, z) for rectangular parallelepiped and random sampling in cylindrical coordinates (r, θ, z) for portion of cylinder. The Figure 4 shows result of the predicted field on the entry port (lower part of the Figure) compared with expected one (upper part of the Figure) for the E_x (left), E_y (middle) and E_z (right). The PINN struggles to fit data from entry port (Port1) as thee neural network converges to the null function.

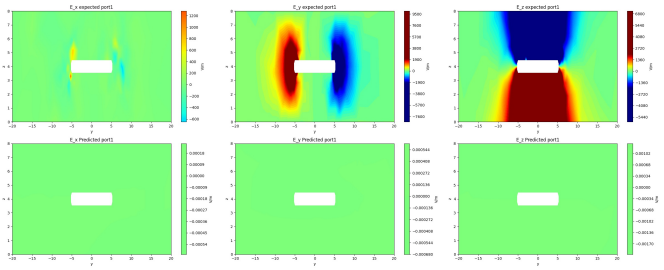


Fig. 4. Electric field on the entry port: expectations(upper part) versus predictions (lower parts) of the neural network after training with gradient sum for respectively (from left to right) E_x, E_y and E_z

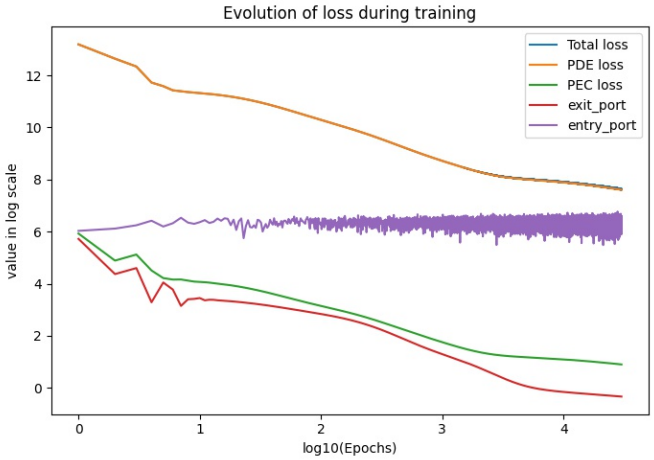


Fig. 5. Loss evolution in log scale for training with gradient sum

By analyzing the training process (Figure 5) we can see that terms of the loss have different range of values. The optimizer will choose the easiest way to reduce the value of the loss even if it means that one term of the loss will be overlooked. In our case the data term $\mathcal{L}_{source}$ (purple) is overlooked which led to the convergence to the null function.

Indeed one can see that the optimization is driven by the loss of the PDE $\mathcal{L}_{PDE}$ in orange and the field converge on a trivial but nonphysical solution of the problem because the boundary condition at the entry port is not respected. Predicting a null field is the most trivial way to respect the Maxwell equations, explaining that the neural network converge towards this solution. A null field also respect the PEC condition and the TE condition, which explains that the decrease of the loss curves for $\mathcal{L}_{PEC}$ in green and $\mathcal{L}_{TE}$ in red is correlated to the decrease of $\mathcal{L}_{PDE}$.

To address this issue, balancing the terms of the loss during the training will be considered in the next section. The choice of those coefficients to balance terms in the loss rely on multi-objective optimization theory.

V. PINN WITH ADAPTIVE TRAINING STRATEGY

A. adaptive training algorithm

A first attempt was made by using the the learning rate annealing method for PINN which is an heuristic algorithm presented by [16]. It is designed to tackle multi-tasks optimization by balancing each gradients so that their contribution will be equal. The motivation under a renormalization of the gradients instead of a renormalization of the loss terms comes from the training procedure of the neural networks. Let $\mathcal{L}$ be the loss for training expressed as a linear combination of the loss terms:

$$\mathcal{L} = \mathcal{L}_{PDE} + \sum_{i=1}^{M} \lambda_i \mathcal{L}_i$$

where $\mathcal{L}_{PDE}$ is the loss of the PDE and $\mathcal{L}_i$ are respectively the different boundary loss terms $\mathcal{L}_{source}$, $\mathcal{L}_{PEC}$ and $\mathcal{L}_{TE}$. The parameters are updated through gradient descent algorithm:

$$\theta_{n+1} = \theta_n - \nabla_\theta \mathcal{L}_{PDE} - \sum_{i=1}^{M} \lambda_i \nabla_\theta \mathcal{L}_i$$

Thus having an equal contribution of each loss term for the parameters update requires for each gradient term to have a similar scale. The learning rate annealing algorithm consists in weighting by using gradient statistics to appropriately balance all terms.
The algorithm is the following:

Algorithm 1 Learning rate annealing for physics-informed neural networks

Consider a physics-informed neural network $f_\theta(\boldsymbol{x})$ with parameters θ and a

$$\mathcal{L}(\theta) := \mathcal{L}_{PDE}(\theta) + \sum_{i=1}^{M} \lambda_i \mathcal{L}_i(\theta)$$

where $\mathcal{L}_{PDE}(\theta)$ denotes the PDE residual loss, the $\mathcal{L}_i(\theta)$ correspond to data-fit terms (e.g., measurements, initial or boundary conditions, etc.), and $\lambda_i = 1, i = 1, \ldots, M$ are free parameters used to balance the interplay between the different loss terms. Then use S steps of a gradient descent algorithm to update the parameters θ as: $n = 1, \ldots, S$
Compute $\hat{\lambda}_i$ by

$$\hat{\lambda}_i = \frac{\max_\theta \left\{ \left| \nabla_\theta \mathcal{L}_{PDE}\left(\theta_n\right) \right| \right\}}{\overline{\left| \nabla_\theta \lambda_i \mathcal{L}_i\left(\theta_n\right) \right|}}, \quad i = 1, \ldots, M,$$

where $\overline{\left| \nabla_\theta \lambda_i \mathcal{L}_i\left(\theta_n\right) \right|}$ denotes the mean of $\left| \nabla_\theta \lambda_i \mathcal{L}_i\left(\theta_n\right) \right|$ with respect to parameters θ. Update the weights λ_i using a moving average of the form

$$\lambda_i = (1-\alpha)\lambda_i + \alpha \hat{\lambda}_i, \quad i = 1, \ldots, M.$$

Update the parameters θ via gradient descent

$$\theta_{n+1} = \theta_n - \eta \nabla_\theta \mathcal{L}_{PDE}\left(\theta_n\right) - \eta \sum_{i=1}^{M} \lambda_i \nabla_\theta \mathcal{L}_i\left(\theta_n\right)$$

The values used are $\eta = 10^{-3}$ and $\alpha = 0.7$.

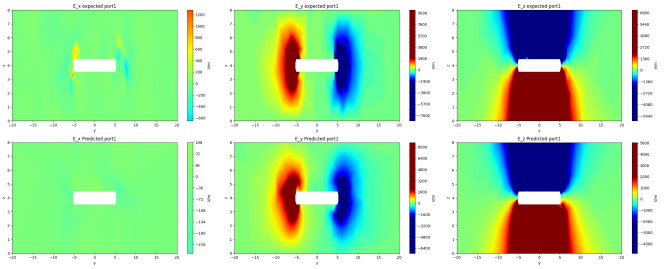


Fig. 6. Electric field on the entry port: expectations(upper part) versus predictions (lower parts) of the neural network after training with learning rate annealing for respectively (from left to right) E_x, E_y and E_z

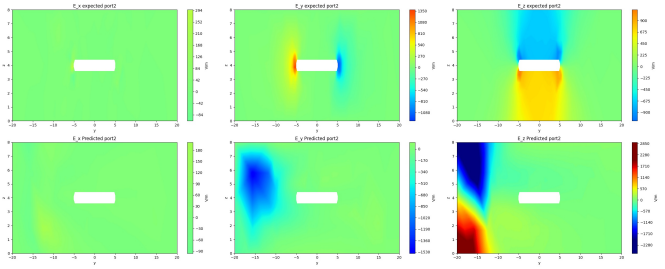


Fig. 7. Electric field on the output port: expectations(upper part) versus predictions (lower parts) of the neural network after training with learning rate annealing for respectively (from left to right) E_x, E_y and E_z

The $\hat{\lambda}_i$ is designed such that each gradient have the same scale as the one for the PDE. λ_i is only partially updated with value $\hat{\lambda}_i$ using the hyperparameter α. The partial update prevents the training from being unstable.

B. Training and results

Using the adaptive trainin procedure, better performances of PINN were noticed on the entry port (Port 1 see Figure 6). the neural net prediction (Figure 6 down) is able to reproduce accurately the reference electric field imposed on the entry port 1 Figure 6 up).

However even by using learning rate annealing method the neural net still struggles to learn accurately the electric field inside the airbox. In Figure 7 electric Field at the exit port shifted to the side. However, one can notice that the TE condition is well respected as the x component of the electric field is approximately zero. $\hat{E}_x$ on the left lower part of the Figure 7 is in good agreement with the reference field E_x on left upper part of the Figure 7. Figure 8 displays the loss curves for the learning of the four different loss terms. When compared to the first attempt with non-adaptive control, the condition at the entry port is learned and decrease especially starting from 90-100 epochs (Figure 8 purple line). The loss curve is also very noisy, showing the difficulty to learn this condition. The three other loss terms have a non monotonic evolution especially with an increase of their value also in 90-100 epochs. The value of these losses are also higher, especially the loss for the partial differential equation in orange. Overall it is much harder to train the neural network in such conditions, also it offers the possibility to converge toward a non-trivial zero field. The Figure 9 shows the evolution of the λ_i values respectively for PEC (green), exit port (red) and entry port (purple) error gradients. One may notice the great difference in order of magnitude for the different values. The exit port has the most important values in the order of 10^8 to 10^{11}. Lambda value for entry port and PEC is of the order of 10^7 to 10^9. The lower the value of the loss term, the greater the value of the lambda. The noisiness of the loss evolution seems to correspond to the one of lambda values. As values of lambda coefficients varies in several order of magnitude during training, it highlights that it is not possible

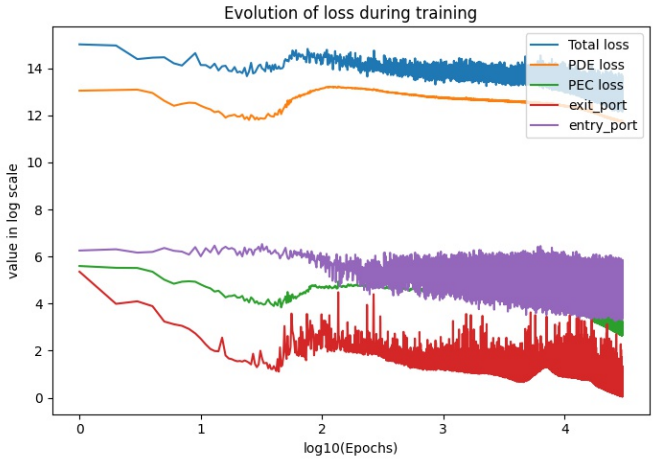


Fig. 8. Loss evolution in log scale for training with learning rate annealing method

to find a fix value of coefficient to balance the gradients during training.

VI. Discussion

From these results we can assert that the PINN is sensitive to the optimization method. Finding an efficient multi-objective optimization algorithm will speed up the convergence and provide us a solution close to the physical problem.
To understand the problem we may focus on the training process. During training process (Figure 8), while the loss related to the entry port decreases the loss related to the PDE increases, meaning that the two terms might be conflicting and the loss related to entry port prevails over the loss related to the PDE. Training PINN is known to be difficult and the use case of the transmission line is particularly challenging as they are three different boundary conditions including two where the value of the electric field is only partially known. A difficulty of training the PINN for the transmission line could reside also in the relatively high derivative order of the partial differential equation and the depth of the neural net. The PINN use backpropagation of the gradients from the output to the input neural network to compute the derivative of the partial differential equation. For the second order derivative, the gradient should be back-propagated twice in the

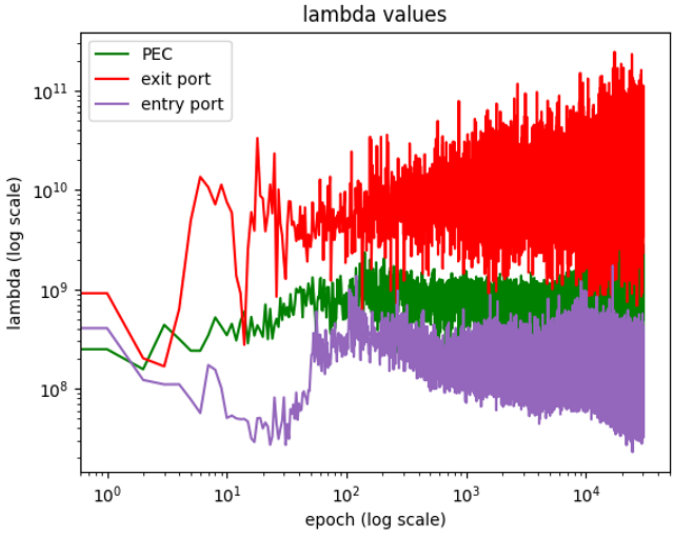


Fig. 9. Lambda evolution in log scale for training with learning rate annealing method

neural network architecture increasing the risk of vanishing or exploding gradient. While the formulation of the neural network parameters optimisation process have been tested in this work, notably using the learning rate annealing gradient algorithm, the selection of the training examples also called collocation points and the architecture of the neural network have been overlooked. Potential future research direction may involve to test residual based adaptive refinements methods for the choice of collocation points [12], [17]. Some works also propose to integrate the boundary conditions as a hard constraint in the neural network architecture [15], suppressing the multi-task optimisation problem for the PINN. Also, the proposition only include Dirichlet boundary conditions. Such method should be adapted if one seek to integrate perfect electrical conductor or transverse electric field conditions in the neural network architecture. An other method proposed to use generative adversial architecture with physics based error term [5]. In such approach the generator aims to generate the solution to a PDE by fooling the discriminator. The PDE constrained is treated as a discriminator input through a physics consistency score, while the boundary condition are treated as examples labelled "true" for the discriminator training. Such approach does not require a balance between a PDE error term and a boundary condition error terms. However as the approach [15], it is well suited for problem with only Dirichlet boundary conditions and can not be applied directly to our use case where there three different kinds of boundary conditions. Concerning the comparison with traditional simulation method, the training of a PINN on the aforementioned use-case is much longer than solving it with standard finite element method, as training the PINN requires several hours as the finite element computation takes only few tens of minutes. The present problem is rather simple for finite element method as it requires only about 3000 elements in the mesh to solve the problem. As highlighted in [8], PINNs approach tends to be slower than finite element methods on simple problem, while they may be more efficient on more complex geometries. The interest of this relatively simple problem is to select the appropriate method for multi-objective optimization before to move to more complex geometries.

VII. Conclusion

In this paper we have presented a first attempt to train a PINN for the simulation of the electric field surronding a transmission line. The field is obtained solving the harmonic Maxwell equations while respecting three different boundary conditions including two non-Dirichlet boundary condition. Training the PINN for transmission lines is a complex multi-task optimisation problem due to the multiplicity of the boundary conditions. A naive approach of the training consist in optimising the neural network parameter using the sum of the four loss terms gradient. In such case, the neural network predict a trivial electric field that is null every where. While this solution is very different from the expected field, it respect three physical constrains on four, namely the partial differential equation, the perfect electrical conductor, the transverse electric field condition. On the other hand the electric field that is impose at the entry port of the transmission line is not respected as the gradient of this error term barely participates to the parameter update. In order to take into account the field at the entry port, we tested an adaptive training strategy that balance the gradients of the different error terms such that they contribute equally to the parameter update. In such case the field at the entry port is well predicted but decreasing the loss on the partial differential equation is much harder. Moreover conflicts between loss terms appears as the decrease of the loss term on the entry port is correlated with an increase of the loss on the partial differential equation. In conclusion, while PINN are promising for solving differential equations, there application to the simulation of transmission line would require to solve issues on the multi-task optimization that is involved. Overcoming this difficulty would allow in the future to fully leverage the advantages of neural networks in the field of simulation such as the generated mesh free solutions to simulation problem with the possibility to transfer knowledge to similar tasks. Design of radio-frequency components for instance for long-range air defence radars could benefit from this method.

VIII. Acknowledgement

The authors thank Jean-Yves Morineau for the simulation of reference data for the electric field and his expertise on problem formulation.

References

[1] Shengze Cai et al. "Physics-informed neural networks (PINNs) for fluid mechanics: A review". In: *Acta Mechanica Sinica* 37.12 (2021), pp. 1727–1738.

[2] Yuyao Chen et al. "Physics-informed neural networks for inverse problems in nano-optics and metamaterials". In: *Optics express* 28.8 (2020), pp. 11618–11633.

[3] Salvatore Cuomo et al. “Scientific machine learning through physics–informed neural networks: where we are and what’s next”. In: *Journal of Scientific Computing* 92.3 (2022), p. 88.

[4] George Cybenko. “Approximation by superpositions of a sigmoidal function”. In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[5] Arka Daw, M Maruf, and Anuj Karpatne. “PID-GAN: A GAN Framework based on a Physics-informed Discriminator for Uncertainty Quantification with Physics”. In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2021, pp. 237–247.

[6] MWMG Dissanayake and Nhan Phan-Thien. “Neural-network-based approximations for solving partial differential equations”. In: *communications in Numerical Methods in Engineering* 10.3 (1994), pp. 195–201.

[7] Zhiwei Fang and Justin Zhan. “Deep physical informed neural networks for metamaterial design”. In: *IEEE Access* 8 (2019), pp. 24506–24513.

[8] Tamara G Grossmann et al. “Can physics-informed neural networks beat the finite element method?” In: *arXiv preprint arXiv:2302.04107* (2023).

[9] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. “Multilayer feedforward networks are universal approximators”. In: *Neural networks* 2.5 (1989), pp. 359–366.

[10] Xiang Huang et al. “Solving partial differential equations with point source based on physics-informed neural networks”. In: *arXiv preprint arXiv:2111.01394* (2021).

[11] George Em Karniadakis et al. “Physics-informed machine learning”. In: *Nature Reviews Physics* 3.6 (2021), pp. 422–440.

[12] Lu Lu et al. “DeepXDE: A deep learning library for solving differential equations”. In: *SIAM review* 63.1 (2021), pp. 208–228.

[13] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations”. In: *Journal of Computational physics* 378 (2019), pp. 686–707.

[14] Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. “Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations”. In: *Science* 367.6481 (2020), pp. 1026–1030.

[15] Luning Sun et al. “Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data”. In: *Computer Methods in Applied Mechanics and Engineering* 361 (2020), p. 112732.

[16] Sifan Wang, Yujun Teng, and Paris Perdikaris. “Understanding and mitigating gradient flow pathologies in physics-informed neural networks”. In: *SIAM Journal on Scientific Computing* 43.5 (2021), A3055–A3081.

[17] Chenxi Wu et al. “A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks”. In: *Computer Methods in Applied Mechanics and Engineering* 403 (2023), p. 115671.

Approche Bayésienne pour l'Estimation des Paramètres de la Dynamique Latérale d'un Véhicule

Fabien Lionti
Université de Côte d'Azur - INRIA
2004 Rte des Lucioles, 06902 Valbonne
Email : fabien.lionti@inria.fr

Nicolas Gutowski
Université d'Angers - LERIA
2 Bd de Lavoisier, 49000 Angers, France
Email : nicolas.gutowski@univ-angers.fr

Sébastien Aubin
Direction Générale de l'Armement - Techniques Terrestres
Rue de la Chédditière, 49460 Montreuil-Juigné
Email : sebastien.aubin@intradef.gouv.fr

Philippe Martinet
Université de Côte d'Azur - INRIA
2004 Rte des Lucioles, 06902 Valbonne
Email : philippe.martinet@inria.fr

***Résumé*—L'estimation des paramètres d'un système dynamique non linéaire constitue un défi majeur dans de nombreux domaines de recherche et d'application. Dans ce contexte, ce papier présente une nouvelle méthode en deux étapes pour estimer les paramètres qui régissent la dynamique latérale d'un véhicule, en tenant compte de la rareté et du bruit des données. La méthode combine le lissage par spline des observations du système avec une approche bayésienne pour estimer les paramètres.**

La première étape de cette méthode consiste à appliquer un lissage par spline aux observations des variables d'état du système. Cette étape permet de filtrer le bruit présent dans les données et d'obtenir des estimations précises des dérivées des variables d'état du système. Ainsi, la méthode proposée permet d'estimer les paramètres directement à partir des équations différentielles décrivant la dynamique du système, sans recourir à des méthodes d'intégration chronophages.

La deuxième étape repose sur l'estimation des paramètres à partir des résidus des équations différentielles, en utilisant une approche bayésienne appelée *likelihood-free ABC-SMC*. Cette approche bayésienne présente plusieurs avantages. Tout d'abord, elle permet de pallier la rareté des données en incorporant des connaissances a priori sur les caractéristiques physiques du véhicule. De plus, elle offre un niveau élevé d'interprétabilité en fournissant une distribution a posteriori sur les paramètres susceptibles d'avoir généré les données observées. L'utilisation de cette nouvelle méthode présente l'avantage de permettre une estimation robuste des paramètres de la dynamique latérale du véhicule, même lorsque les données sont limitées et bruitées.

I. Introduction

L'estimation des paramètres de systèmes dynamiques non linéaires est un problème de recherche largement exploré [1]. Une approche classique pour résoudre ce problème repose sur la minimisation d'une fonction objectif qui mesure l'écart entre les prédictions du modèle et les données observées, en utilisant des méthodes de descente de gradient [2]. Cependant, cette approche présente certaines limitations, notamment la possibilité de converger vers des minimums locaux plutôt que le minimum global, ainsi que la sensibilité du système aux conditions initiales.

Afin de surmonter ces limitations, diverses techniques ont été proposées. Par exemple, l'utilisation d'algorithmes d'optimisation globale tels que les algorithmes génétiques [3], le recuit simulé [4] ou encore l'optimisation par essaim de particules [5] permet de rechercher plus efficacement l'espace des paramètres à la recherche du minimum global. Les approches mentionnées sont limitées par le nombre de paramètres à estimer. En effet, lorsque le nombre de paramètres augmente, les temps de calcul augmentent eux aussi et de manière exponentielle. De plus, l'utilisation de méthodes d'intégration, comme la méthode de *Runge-Kutta* [6], pour comparer les prédictions aux observations entraîne des temps de calcul significatifs, car il est nécessaire d'intégrer le système pour chaque condition initiale étudiée.

Dans le cas d'observations bruitées, il peut être difficile de déterminer avec précision les conditions initiales d'intégration, ce qui peut entraîner des estimations biaisées lorsque le système est sensible à celles-ci. Afin de réduire cette sensibilité et éliminer l'étape d'intégration, une approche consiste en le lissage des observations par un spline [6] à partir duquel une estimation des dérivées des observations du système est obtenue. Il devient alors possible d'estimer les paramètres à partir du résidu de l'équation différentielle. Ce principe a été récemment étendu aux réseaux de neurones. Les Physics-Informed Neural Networks (PINN) [7] permettent d'interpoler des observations correspondant à un processus décrit par des équations aux dérivées partielles (EDP), tout en estimant les paramètres de ces équations. Cela est réalisé en s'appuyant sur le mécanisme de différenciation automatique afin d'optimiser les paramètres par descente de gradient à partir de l'erreur quadratique du résidu de l'EDP.

Dans le domaine de l'estimation robuste, les approches mentionnées précédemment sont limitées par leur incapacité à fournir une interprétation probabiliste des paramètres estimés. Cependant, des travaux récents ont cherché à combler cette lacune en incorporant des connaissances sur les distributions a priori des paramètres. Ces travaux s'inscrivent dans le cadre bayésien, offrant ainsi deux avantages : restreindre l'espace de recherche des paramètres grâce aux a priori spécifiés, tout en permettant une meilleure interprétabilité grâce à l'obtention

d'une distribution a posteriori sur les paramètres. L'approche décrite par [8] et [9] consiste à utiliser un processus gaussien pour interpoler et modéliser l'incertitude liée au bruit des observations d'un système dynamique. Étant donné que la dérivée d'un processus gaussien reste un processus gaussien [10], il est possible de définir une fonction de vraisemblance en combinant ce processus avec un schéma d'échantillonnage tel que l'algorithme de Metropolis-Hastings [11] ou de Gibbs [12]. Ce schéma permet d'évaluer une distribution a posteriori sur les paramètres en supposant une erreur normale entre les dérivées observées et celles prédites par le processus gaussien.

La difficulté de ces méthodes réside dans le paramétrage exigeant des méthodes d'échantillonnage, tel que le choix du nombre d'échantillons afin d'approximer la distribution a posteriori, ou le choix de la proposition de transition. Le choix des propositions de transition peut avoir un impact important sur l'efficacité et la performance des méthodes. Des propositions de transition mal spécifiées peuvent entraîner un taux d'acceptation faible ou une exploration inefficace de l'espace des échantillons. Il est souvent nécessaire de tester et d'ajuster différentes propositions de transition pour optimiser les performances des méthodes. De plus, la convergence des méthodes d'échantillonnage est une préoccupation majeure. Il est essentiel que les échantillons générés convergent vers la distribution cible. Cependant, dans certains cas, les méthodes peuvent converger lentement, en particulier lorsque la distribution cible présente des propriétés complexes telles que des modes multiples ou des régions de faible probabilité.

En parallèle des méthodes *likelihood-free* ont été mises au point afin de directement approximer la distribution a posteriori sans à avoir la nécessité de définir une fonction de vraisemblance qui est souvent difficile à établir ou coûteuse à évaluer. Dans ce contexte, la méthode *Approximate Bayes Computation (ABC)* [13] permet de fournir une approximation de la distribution a posteriori des paramètres en utilisant un échantillonnage basé sur la simulation. Cette approche repose sur un filtrage des paramètres qui auraient pu générer les données observées par l'utilisation d'une mesure de similarité entre les données simulées et les données observées. L'avantage de cette méthode réside dans sa simplicité d'utilisation car elle ne requiert pas de paramétrages complexes. Afin d'améliorer l'efficacité de *ABC*, l'approche *ABC-SMC* [14] pour *Sequential Monte Carlo*, permet d'augmenter l'efficacité de l'échantillonnage en minimisant le nombre de rejets grâce à un échantillonnage préférentiel qui se concentre sur les zones de l'espace des paramètres les plus probables par rapport aux données observées.

L'état de l'art effectué n'a pas permis d'identifier une application en lien avec l'estimation de paramètres dans le cadre d'un système dynamique non-linéaire de véhicule. Cependant, ces méthodes ont démontré leur efficacité dans l'estimation de paramètres de systèmes dynamiques liés à des processus biologiques [15] [16] [17]. L'approche proposée est évalué sur des données de simulation et s'appuie sur un modèle bicyclette intégrant un modèle de pneumatiques de Pacejka [18]. La problématique questionnée dans cet article concerne l'estimation des coefficients de Pacejka, ainsi que certains paramètres physiques du modèle tels que la position du centre de gravité et le moment d'inertie du véhicule. La méthode repose sur un lissage des observations par spline et l'application de l'approche *ABC-SMC* afin d'estimer la distribution a posteriori des paramètres. Ces travaux s'inscrivent plus globalement dans une volonté d'obtenir une modélisation de véhicules militaires terrestres, permettant ainsi d'étudier leur stabilité et de garantir un comportement sécuritaire. L'utilisation d'un modèle numérique permettrait de réduire les coûts associés aux campagnes d'essais, nécessitant aujourd'hui de tester le véhicule dans de nombreux environnements et sous différentes configurations.

Cet article est organisé comme suit : la section II décrit de manière générale le cadre bayésien appliqué à l'estimation de paramètres. La section III expose le principe de l'approche *ABC* ainsi que les différents paramètres associées. La section IV formalise le modèle dynamique utilisé. La section V décrit notre méthode. La section VI analyse les résultats d'expérimentations effectuées. Enfin, la dernière section VII conclut et expose les futurs travaux envisagés.

II. Cadre Bayésien

Le cadre bayésien offre une approche idéale pour l'inférence des paramètres des systèmes dynamiques lorsque plusieurs modèles peuvent potentiellement correspondre aux données observées. Dans cette approche, l'ensemble de données $\mathcal{D}$ est définit comme étant la représentation des observations d'un système dynamique tandis que $\theta = \{\theta_1, \theta_2, ..., \theta_n\}$ correspond à l'ensemble des paramètres candidats pour un système d'équations différentielles (ED).

L'équation de Bayes permet de calculer la probabilité a posteriori des paramètres θ étant données les données $\mathcal{D}$, comme suit :

$$P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta) \cdot P(\theta)}{P(\mathcal{D})} \tag{1}$$

où :

— $P(\theta|\mathcal{D})$ représente la probabilité a posteriori des paramètres θ étant données les données $\mathcal{D}$.
— $P(\mathcal{D}|\theta)$ est la vraisemblance des données $\mathcal{D}$ étant données les paramètres θ.
— $P(\theta)$ est la probabilité a priori des paramètres θ.
— $P(\mathcal{D})$ est la probabilité marginale des données $\mathcal{D}$.

L'équation de Bayes permet de mettre à jour nos connaissances sur les paramètres du modèle en fonction des données observées et offre un cadre systématique pour incorporer des connaissances a priori dans le processus d'inférence.

Le cadre bayésien, dans sa forme initiale, nécessite d'établir la fonction de vraisemblance $P(\mathcal{D}|\theta)$, ce qui pose plusieurs défis. Les systèmes dynamiques non linéaires sont modélisés par des ED complexes. Établir une fonction de vraisemblance analytique peut être difficile, voire impossible. Cela peut nécessiter l'utilisation de techniques d'approximation ou de méthodes numériques pour estimer la vraisemblance. La

probabilité marginale $P(\mathcal{D})$ est calculée en intégrant la vraisemblance $P(\mathcal{D}|\theta)$ sur l'ensemble de l'espace des paramètres θ, pondéré par la probabilité a priori $P(\theta)$:

$$P(\mathcal{D}) = \int P(\mathcal{D}|\theta) \cdot P(\theta)\ d\theta \tag{2}$$

Dans de nombreux cas, cette intégrale n'a pas de forme analytique connue et ne peut pas être calculée directement. Cela est particulièrement vrai lorsque le modèle est non-linéaire et que la dimension des paramètres est élevée.

III. Approximate Bayes Computation

La méthode *ABC* (Algorithme 1) permet d'estimer la distribution a posteriori $P(\theta|\mathcal{D})$ sans avoir à spécifier la fonction de vraisemblance et sans avoir à calculer la probabilité marginale des données. Le seul pré-requis est de pouvoir simuler des données $\mathcal{D}^*$ à partir d'un modèle correspondant à différentes paramétrisations de θ.

Algorithm 1: Algorithme *ABC*

Data: Données observées $\mathcal{D}$, taille de l'échantillon N, seuil d'acceptation ϵ
Result: Approximation de la distribution a posteriori
while $i \leq N$ **do**
  Génération d'un échantillon $\tilde{\theta}$ à partir de la distribution a priori;
  Génération de données simulées $\mathcal{D}^*$ à partir de $\tilde{\theta}$;
  Calcul de la distance $\rho(\mathcal{D}, \mathcal{D}^*)$;
  if $\rho < \epsilon$ **then**
    $\theta^i \leftarrow \tilde{\theta}$;
    $i = i + 1$;
  end
end

Cette approche reste inefficace dans le choix des échantillons θ au cours des itérations, car elle explore l'espace des paramètres sans favoriser les régions de l'espace les plus susceptibles d'engendrer une acceptation de l'échantillon. L'approche *ABC-SMC* (Algorithme 2) permet d'améliorer l'efficacité de l'échantillonnage en s'appuyant sur le principe d'utiliser efficacement les informations antérieures. Dans l'*ABC-SMC*, les échantillons de l'itération précédente sont utilisés pour générer de nouveaux échantillons. Cela permet de tirer parti des informations déjà acquises sur la distribution a posteriori et de concentrer les efforts de calcul sur les régions les plus prometteuses de l'espace des paramètres. Ainsi, en améliorant la qualité de l'échantillonnage, il est possible d'obtenir une amélioration progressive de l'approximation de la distribution a posteriori en utilisant une séquence de seuils décroissants de plus en plus stricts. La génération de nouveaux échantillons repose sur un mécanisme d'échantillonnage par importance des précédents échantillons en pondérant les chances de tirage. Chaque échantillon tiré est alors perturbé par une fonction de perturbation K, souvent uniforme ou gaussienne, qui permet de modéliser la probabilité de transition $P(\theta|\tilde{\theta}_{t-1})$. Elle peut être paramétrisée pour contrôler l'amplitude des perturbations et ainsi influencer la diversité et l'exploration de l'espace des paramètres. Après génération de nouveaux échantillons, ils sont évalués à l'aide de la mesure d'adéquation ρ et les échantillons générant les données $\mathcal{D}^*$ les plus proches des observations $\mathcal{D}$ sont sélectionnés selon le seuil ϵ_t pour la prochaine itération du processus d'échantillonnage. Cette étape itérative est répétée jusqu'à ce que la convergence de la distribution a posteriori soit atteinte ou que d'autres critères de convergence soient satisfaits.

Algorithm 2: Algorithme *ABC-SMC*

Data: Données observées $\mathcal{D}$, nombre d'itérations T, taille de l'échantillon N, seuils $\epsilon_1 > \epsilon_2 > ... > \epsilon_T$
Result: Approximation de la distribution a posteriori
Initialisation de l'échantillon $\theta^1, \theta^2, ..., \theta^N$;
for $t \leftarrow 1$ **to** T **do**
  while $i \leq N$ **do**
    if $t = 1$ **then**
      $\tilde{\theta} \leftarrow P(\theta)$;
    end
    else
      Tirer un échantillon $\tilde{\theta}_{t-1}$ à partir de l'ensemble des échantillons de l'itération précédente θ_{t-1} pondéré par w;
      $\tilde{\theta} \leftarrow P(\theta|\tilde{\theta}_{t-1})$;
    end
    if $P(\tilde{\theta}) \neq 0$ **then**
      Génération de données simulées $\mathcal{D}^*$ à partir de $\tilde{\theta}$;
      Calcul de la distance $\rho(\mathcal{D}, \mathcal{D}^*)$;
      if $\rho < \epsilon_t$ **then**
        $\theta_t^i \leftarrow \tilde{\theta}$;
        if $t \neq 1$ **then**
          $w_t^i = \frac{P(\theta_t^i)}{\sum_{j=1}^{N} w_{t-1}^i K(\theta_t^i, \theta_{t-1}^j)}$
        end
        else
          $w^i = 1$
        end
        $i = i + 1$;
      end
    end
  end
  for $j \leftarrow 1$ **to** N **do**
    $w_t^j = \frac{w_t^j}{\sum_{i=1}^{N} w_t^i}$
  end
end

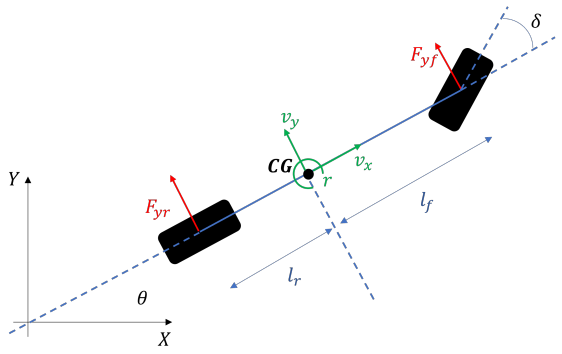


FIGURE 1 – Modèle bicyclette.

IV. MODÉLISATION DE LA DYNAMIQUE LATÉRALE D'UN VÉHICULE

A. Modèle bicyclette à deux dégrées de liberté

Le modèle de bicyclette à deux degrés de liberté est une représentation simplifiée de la dynamique latérale d'un véhicule. Il est largement utilisé dans le domaine de la dynamique des véhicules pour analyser le comportement de conduite, la stabilité et la maniabilité des véhicules. En appliquant le principe fondamental de dynamique au modèle des forces décrit Figure 1 on obtient le système d'ED suivant :

$$\begin{cases} \dot{V}_y = \frac{F_{yf}}{m}\cos\delta_f + \frac{F_{yr}}{m} - v_x r \\ \dot{r} = \frac{L_f}{I_z}F_{yf}\cos\delta_f - \frac{L_r}{I_z}F_{yr} \end{cases} \tag{3}$$

La vitesse de lacet du véhicule est représentée par r, la distance des essieux avant par rapport au centre de gravité (CG) est représentée par l_f, la distance des essieux arrière par rapport au CG est représentée par l_r, et l'inertie en lacet du véhicule est représentée par I_z. Les forces latérales exercées par les pneus avant et arrière du véhicule sont modélisées à l'aide du modèle non-linéaire de Pacejka [18].

B. Modèle de Pacejka

Le modèle de Pacejka est un modèle dit hybride dans la mesure où il a été établi à partir de lois physiques en supposant une répartition parabolique des forces sur la zone de contact pneu/sol mais en se fondant également sur des observations phénoménologiques obtenues à partir de résultats d'essais. Ainsi, il s'agit d'un compromis entre un temps de calcul assez rapide et une précision acceptable. Le modèle de Pacejka, consiste à modéliser les forces exercées par la route sur les pneus.

$$F_{yf} = D\cdot\sin\Big(C\cdot\arctan\Big(B\cdot\big(\alpha_f - E\cdot(\alpha_f - \arctan\big(B\cdot\alpha_f - \arctan\left(B\alpha_f\right)\big)\big)\Big)\Big)\Big) \tag{4}$$

$$F_{yr} = D\cdot\sin\Big(C\cdot\arctan\Big(B\cdot\big(\alpha_r - E\cdot(\alpha_r - \arctan\big(B\cdot\alpha_r - \arctan\left(B\alpha_r\right)\big)\big)\Big)\Big)\Big) \tag{5}$$

B est appelé le coefficient de rigidité, C le coefficient de forme, D est la valeur maximale atteinte par F_y, E est le facteur de courbure.

Le modèle de Pacejka prend en entrée la dérive des pneumatiques avant α_f et arrière α_r, correspondant à la différence entre le vecteur vitesse réel v des pneumatiques et v_y (Figure 2). Cet angle de dérive est influencé par des facteurs tels que la vitesse latérale du véhicule, la vitesse de rotation, la distance entre l'essieu avant/arrière et le centre de gravité du véhicule, ainsi que l'angle de braquage des roues avant.

$$\begin{cases} \alpha_f = \frac{v_y + L_f r}{v_x} - \delta_f \\ \alpha_r = \frac{v_y - L_r r}{v_x} \end{cases} \tag{6}$$

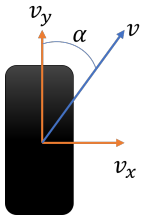


FIGURE 2 – Dérive latérale

V. MÉTHODE EMPLOYÉE

A. Génération des données

À partir du modèle décrit précédemment les données sont utilisées pour l'estimation des paramètres $\theta = \{B, C, D, E, l_f, l_r, I_z\}$. Le système d'ED est intégré par méthode de Runge-Kutta avec une période d'échantillonnage de 0.01 secondes pour une durée de 60 secondes. Un bruit gaussien est additionné aux trajectoires obtenues de moyenne nulle et de variance $\sigma^2_{vy} = 2.5e^{-3}$ à v_y et $\sigma^2_r = 6.25e^{-6}$ à r. les paramètres d'intégrations utilisés sont les suivants :

Paramètre	B	C	D	E	l_f	l_r	I_z	m	vx
Valeur	3	2	1	1	1.5	3	1.9	1500	30

B. Paramétrages de la méthode

La méthode développée (Figure 3) se déroule en deux étapes. Dans un premier temps, les observations obtenues sont lissées pour les variables d'état du système v_y et r à l'aide d'un spline cubique. Afin de filtrer le bruit des observations, le spline est régularisé en pénalisant sa courbure de manière à ne pas interpoler le bruit des observations. Deux fonctions $S_{vy}(x)$ et $S_r(x)$ sont obtenues permettant d'approximer les dérivées des deux variables d'état du système $\dot{S}_{vy}(x)$ et $\dot{S}_r(x)$ malgré le bruit initialement présent dans les observations. $x \in X$ correspond à la position des observations au cours du temps, $X = \{0.01, 0.02, ..., 60.0\}$.

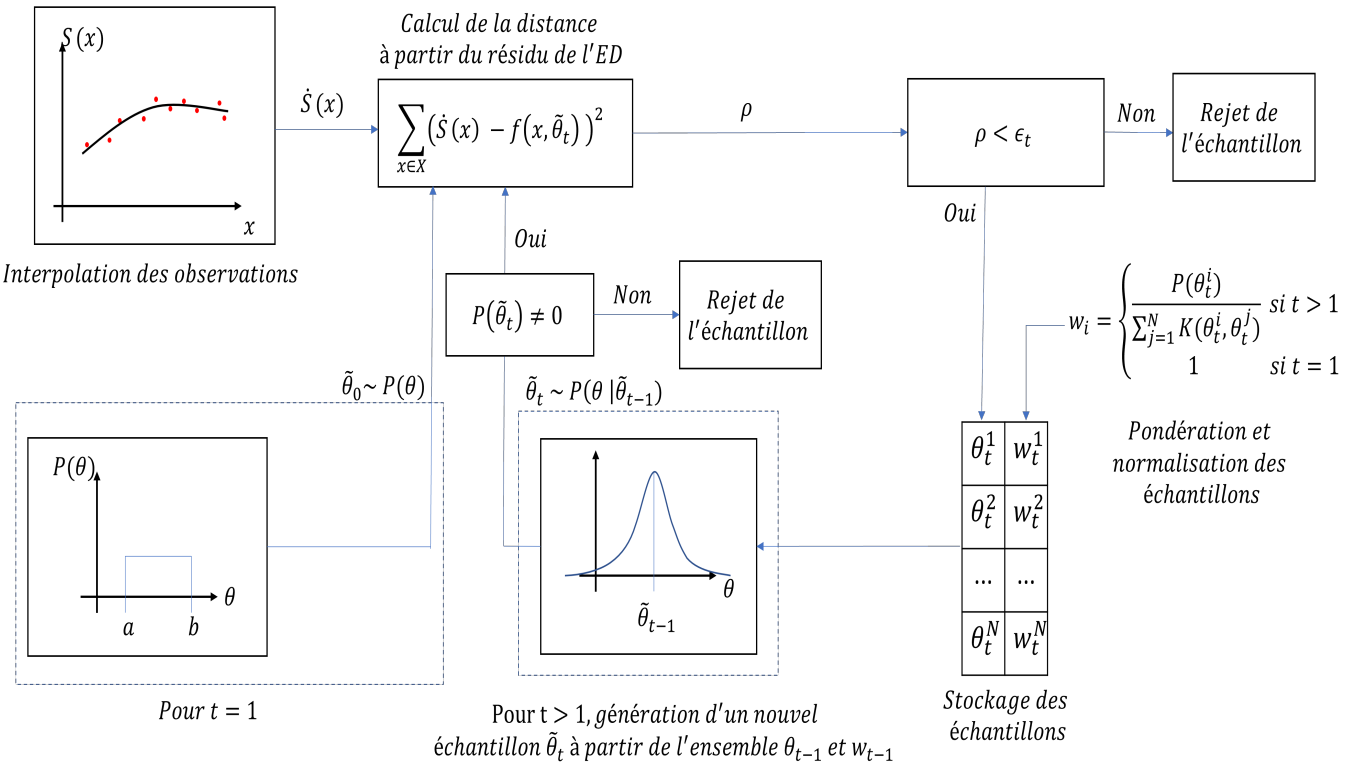


FIGURE 3 – Description générale de la méthode : La fonction $S(x)$ correspond à une interpolation réalisée par spline cubique sur les observations d'une variable d'état. Les différentes étapes au cours des itérations t de l'algorithme *ABC-SMC* y sont représentées. $f(x, \theta_t)$ représente une équation différentielle paramétrée par θ_t

.

Le résidu de l'ED décrivant la dynamique du système est définit de la manière suivante :

$$\begin{cases} \dot{S}_{vy}(x) - \frac{F_{yf}}{m}\cos\delta_f + \frac{F_{yr}}{m} - v_x S_r(x) = 0 \\ \dot{S}_r(x) - \frac{L_f}{I_z}F_{yf}\cos\delta_f - \frac{L_r}{I_z}F_{yr} = 0 \end{cases} \tag{7}$$

Une approximation de la solution de ce système d'ED est donnée pour :

$$\hat{\theta} = \arg\min_{\theta\in\Theta} \rho(\theta) \tag{8}$$

avec :

$$\rho(\theta) = \frac{1}{|X|}\sum_{x\in X}\left(\left(\dot{S}_{vy}(x) - \frac{F_{yf}}{m}\cos\delta_f + \frac{F_{yr}}{m} - v_x S_r(x)\right)^2 + \left(\dot{S}_r(x) - \frac{L_f}{I_z}F_{yf}\cos\delta_f - \frac{L_r}{I_z}F_{yr}\right)^2\right) \tag{9}$$

La valeur de $\rho(\theta)$ correspond à une métrique d'adéquation entre les paramètres θ et la solution de l'ED. $\rho(\theta)$ est utilisée en tant que distance au sein de l'algorithme *ABC-SMC* de manière à rejeter les paramètres qui ont peu de chances de correspondre à une solution du système d'ED.

Afin de contraindre la recherche des solutions de l'ED, un a priori uniforme $P(\theta) = unif(a, b)$ est positionné sur chacun des 7 paramètres à estimer :

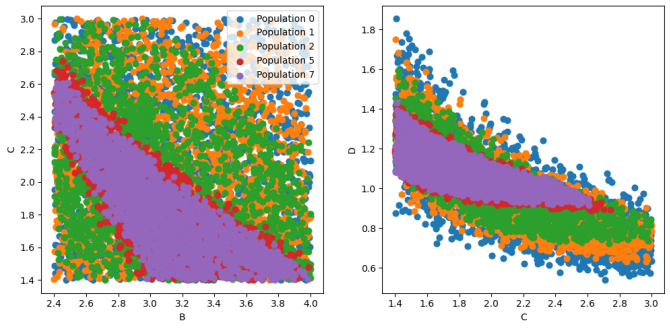
Paramètre	B	C	D	E	l_f	l_r	I_z
a	2.4	1.4	0.4	0.4	0.9	2.4	1.3
b	4	3	2	2	2.5	4	2.9

Un nombre d'itérations $T = 8$ est spécifié, pour chacune des itérations $N = 2000$ échantillons sont générés. Afin de filtrer les échantillons au cours de chacune des itérations, les seuils utilisés sont les suivants $\epsilon_t = \{0.5, 0.3, 0.15, 0.05, 0.025, 0.02, 0.015, 0.0125\}$

La fonction de perturbation $P(\theta|\tilde{\theta}_{t-1}) = \mathcal{N}(\mu, \sigma^2)$ utilisée est une Gaussienne de variance $\sigma^2 = 0.01$ et $\mu = \tilde{\theta}_{t-1}$

VI. RÉSULTATS OBTENUS

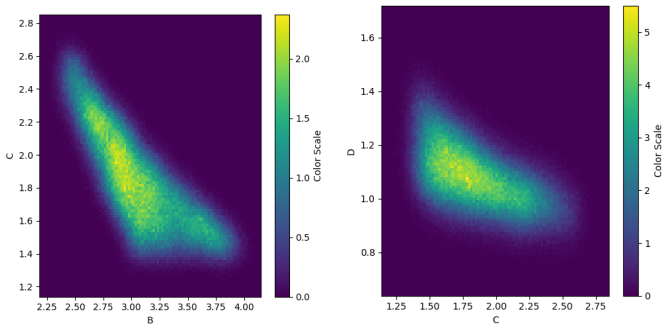
La Figure 4 permet d'observer la convergence des échantillons au cours des itérations de l'algorithme *ABC-SMC*. Ces échantillons explorent l'espace des paramètres contraint par la distribution a priori $P(\theta)$ et convergent progressivement vers une estimation de la densité de probabilité a posteriori $P(\theta|\mathcal{D})$, qui est approximée par les échantillons de la septième population.



(a) Entre B et C (b) Entre C et D

(c) Entre D et E (d) Entre lf et lr

FIGURE 4 – Diagrammes de paires montrant la convergence des échantillons vers la distribution a posteriori.



(a) Entre B et C (b) Entre C et D

(c) Entre D et E (d) Entre lf et lr

FIGURE 5 – Diagramme de paires issu d'une estimation par densité à noyaux sur les échantillons obtenus

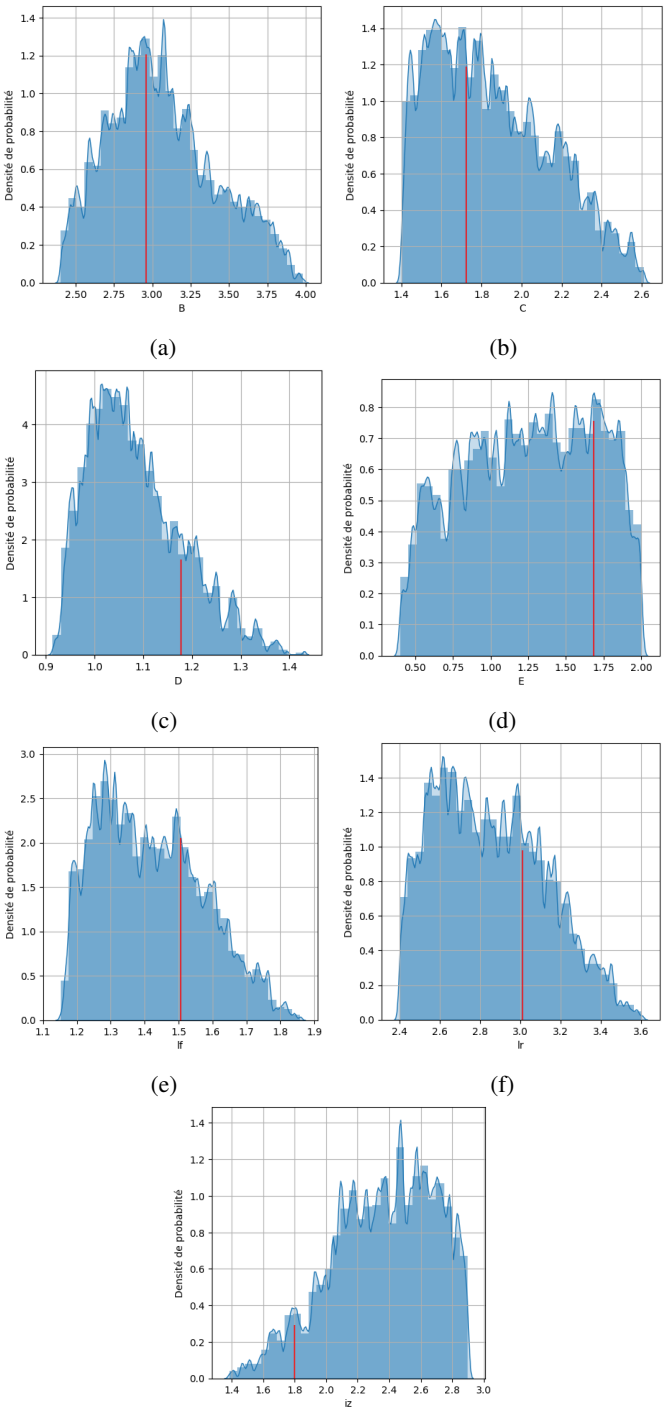
L'approche *ABC-SMC*, s'inscrit dans une démarche globale grâce à la distribution de probabilité a posteriori obtenue. Cette distribution fournit un support pour analyser d'un point de vue statistique la sensibilité des différents paramètres du système vis-à-vis des observations utilisées. Afin d'illustrer cette démarche, les résultats présentés dans cette section s'intéressent dans un premier temps à l'analyse de la sensibilité des paramètres du modèle de la dynamique latérale d'un véhicule. Afin de visualiser la distribution finale obtenue, la Figure 6 et la Figure 10 (présente en annexe) montrent une estimation de la densité de probabilité a posteriori par densité de noyaux basée sur les échantillons finaux obtenus. Les paramètres présentant une variance élevée indiquent une faible sensibilité du système à leur égard, ce qui signifie que ces dimensions de l'espace de recherche ont une influence limitée sur les chances qu'un échantillon soit accepté ou rejeté. La Figure 5 se focalise sur les paires de paramètres faisant apparaître des corrélations indiquant une sensibilité du système à ceux-ci. Ces graphiques exposent des corrélations entre lf et lr, faisant référence à la longueur du véhicule et à la position du centre de gravité. Ils montrent également des corrélations entre les paramètres C et D, ainsi qu'entre C et B. En complément de ces corrélations, le paramètre D présente un faible écart-type, ce qui correspond, dans le modèle de Pacejka, à la force maximale pouvant être générée par les pneumatiques sur le sol. Cette sensibilité est cohérente avec les données simulées, qui représentent des trajectoires en courbe avec une vitesse élevée. L'estimation de E se révèle éloignée de celui utilisé pour simuler les données. La Figure 6 montre une dispersion importante dans la distribution de ce paramètre. Cette dispersion s'explique par le fait que le paramètre E représente le facteur de courbure atteint au pic d'adhérence. Une estimation précise de ce facteur de courbure nécessiterait de soumettre le véhicule à des dynamiques plus poussées, engendrant des situations de glissement potentiellement à risque. L'analyse des distributions de probabilité obtenues offre une interprétation sur l'observabilité des paramètres du système.

Parmi tous les échantillons obtenus lors de la dernière itération de l'algorithme *ABC-SMC*, l'échantillon $\hat{\theta}$ ayant généré une mesure $\rho(\hat{\theta})$ minimale est retenu :

Paramètre θ	B	C	D	E	l_f	l_r	I_z
$\hat{\theta}$	2.961	1.723	1.176	1.685	1.507	3.009	1.798
$Ecart-type$	0.348	0.291	0.095	0.427	0.153	0.269	0.327

Les Figures 7 et 8 permettent de visualiser l'interpolation des observations réalisées avec le spline cubique. L'intégration du système avec pour valeur de paramètres $\hat{\theta}$ permet d'obtenir une solution très proche de la trajectoire modélisée par le spline. L'erreur RMSE est calculé entre les trajectoires obtenues par intégration du système avec $\hat{\theta}$ et les observations :

Paramètre	v_y	r
$\hat{\theta}$	0.0398	0.0020
θ^*	0.0397	0.0020



(a) (b) (c) (d) (e) (f) (g)

FIGURE 6 – Distributions de probabilités obtenues pour chacun des 7 paramètres estimés. La courbe représente l'approximation de la distribution réalisée avec un estimation par densité de noyaux. En rouge est tracée la valeur de $\hat{\theta}$ pour chacun des paramètres

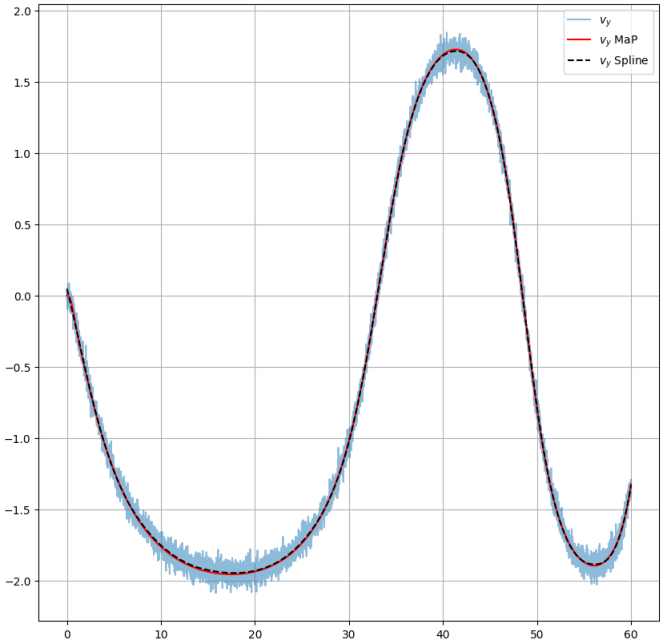


FIGURE 7 – Évolution de $v_y(x)$. En rouge la trajectoire suivis pour la paramétrisation $\hat{\theta}$, bleu par θ^*, en pointillé l'interpolation $S_{vy}(x)$

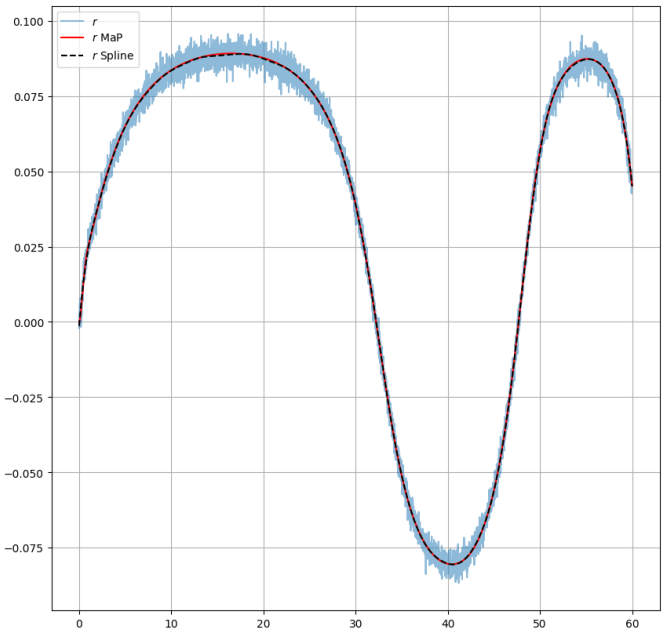


FIGURE 8 – Évolution de $r(x)$. En rouge la trajectoire suivis pour la paramétrisation $\hat{\theta}$, bleu par θ^*, en pointillé l'interpolation $S_r(x)$

L'erreur RMSE obtenue à la suite de l'intégration générée par les paramètres θ^* représente notre référence de base. Cette erreur correspond au bruit additionnel ajouté à chacune des variables d'état. En comparant l'erreur RMSE entre les paramètres $\hat{\theta}$ et celle obtenue avec θ^*, nous constatons que les paramètres estimés par notre méthode génèrent une erreur RMSE similaire à celle de θ^* pour chacune des deux variables d'état. Cela démontre que notre méthode a été capable de filtrer le bruit des observations et d'identifier un ensemble de paramètres qui auraient pu être à l'origine des données observées.

En raison de la faible quantité de données et de la faible sensibilité du système à certains des paramètres, les densités de probabilité a posteriori pour certains des paramètres sont relativement plates et présentent de nombreux modes (Figure 6). Ces modes correspondent à des minima locaux par rapport à la métrique ρ établie. Les positions marquées en rouge au sein de chacune de ces distributions correspondent à la valeur de chacune des composantes de $\hat{\theta}$. Ces valeurs prises indépendamment ne permettent pas d'évaluer visuellement le pic correspondant au maximum de la distribution a posteriori, ce qui explique pourquoi certaines valeurs semblent éloignées du pic d'amplitude maximale.

VII. Conclusion

La méthode proposée permet d'obtenir une approximation de la distribution a posteriori des paramètres liés à la dynamique latérale d'un véhicule. Le lissage par spline permet de filtrer le bruit présent dans les observations et de travailler directement sur une estimation des dérivées des variables d'état du système. Cela ouvre la voie à l'utilisation de l'approche bayésienne *ABC-SMC* en utilisant une métrique de distance basée sur le résidu de l'équation différentielle. En utilisant cette métrique, nous pouvons nous dispenser des temps de calcul liés à l'intégration du système, en favorisant la génération d'un plus grand nombre d'échantillons avec un seuil de rejet plus strict, ce qui permet d'obtenir une meilleure convergence vers la densité de probabilité a posteriori.

L'apport de connaissances a priori permet de compenser la faible quantité d'observations disponibles en restreignant l'espace d'exploration des paramètres à estimer. L'obtention d'une distribution de probabilité a posteriori permet d'étudier la sensibilité du système et la nature des relations entre ses différents paramètres, offrant ainsi une interprétabilité des solutions obtenues.

Nous partons de l'hypothèse que le modèle utilisé pour générer les observations est similaire au modèle pour lequel nous cherchons à estimer les paramètres. Pour renforcer la robustesse de notre approche, nous prévoyons d'estimer les paramètres d'un système d'équations différentielles tout en tenant compte simultanément des divergences entre le modèle choisi et les observations.

VIII. Remerciement

Ces recherche sont soutenues financièrement par le Ministère des Armées via l'Agence de l'Innovation de Défense (AID) et par l'Institut Nationale de Recherche en Informatique et Automatique (INRIA).

RÉFÉRENCES

[1] N. K. Sinha, "System identification - theory for the user : Lennart ljung." *Autom.*, vol. 25, no. 3, pp. 475–476, 1989. [Online]. Available : http://dblp.uni-trier.de/db/journals/automatica/automatica25.html#Sinha89

[2] L. T. Biegler, J. J. Damiano, and G. E. Blau, "Nonlinear parameter estimation : A case study comparison," *AIChE Journal*, vol. 32, no. 1, pp. 29–45, 1986. [Online]. Available : https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690320105

[3] S. Wang and X. Xu, "Parameter estimation of internal thermal mass of building dynamic models using genetic algorithm," *Energy Conversion and Management*, vol. 47, no. 13, pp. 1927–1941, 2006. [Online]. Available : https://www.sciencedirect.com/science/article/pii/S0196890405002311

[4] A. Eftaxias, J. Font, A. Fortuny, A. Fabregat, and F. Stüber, "Nonlinear kinetic parameter estimation using simulated annealing," *Computers Chemical Engineering*, vol. 26, no. 12, pp. 1725–1733, 2002. [Online]. Available : https://www.sciencedirect.com/science/article/pii/S0098135402001564

[5] M. Schwaab, E. C. Biscaia, Jr., J. L. Monteiro, and J. C. Pinto, "Nonlinear parameter estimation through particle swarm optimization," *Chemical Engineering Science*, vol. 63, no. 6, pp. 1542–1552, 2008. [Online]. Available : https://www.sciencedirect.com/science/article/pii/S0009250907008755

[6] U. M. Ascher and L. R. Petzold, *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, 1st ed. USA : Society for Industrial and Applied Mathematics, 1998.

[7] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks : A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019. [Online]. Available : https://www.sciencedirect.com/science/article/pii/S0021999118307125

[8] D. Barber and Y. Wang, "Gaussian processes for bayesian estimation in ordinary differential equations," in *International Conference on Machine Learning*, 2014.

[9] B. Calderhead, M. Girolami, and N. Lawrence, "Accelerating bayesian inference over nonlinear differential equations with gaussian processes," in *Advances in Neural Information Processing Systems*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds., vol. 21. Curran Associates, Inc., 2008. [Online]. Available : https://proceedings.neurips.cc/paper_files/paper/2008/file/07563a3fe3bbe7e3ba84431ad9d055af-Paper.pdf

[10] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning.*, ser. Adaptive computation and machine learning. MIT Press, 2006.

[11] S. Chib and E. Greenberg, "Understanding the metropolis-hastings algorithm," *The American Statistician*, vol. 49, no. 4, pp. 327–335, 1995. [Online]. Available : http://www.jstor.org/stable/2684568

[12] G. Casella and E. I. George, "Explaining the gibbs sampler," *The American Statistician*, vol. 46, no. 3, pp. 167–174, 1992. [Online]. Available : http://dx.doi.org/10.2307/2685208

[13] J.-M. Marin, P. Pudlo, C. P. Robert, and R. Ryder, "Approximate Bayesian computational methods," *Statistics and Computing*, vol. 22, no. 6, Nov. 2012. [Online]. Available : https://hal.science/hal-00567240

[14] T. Toni, D. Welch, N. Strelkowa, A. Ipsen, and M. P. Stumpf, "Approximate bayesian computation scheme for parameter inference and model selection in dynamical systems," *Journal of the Royal Society Interface*, vol. 6, no. 31, pp. 187–202, 2009.

[15] J. Liepe, P. Kirk, S. Filippi, T. Toni, C. P. Barnes, and M. P. H. Stumpf, "A framework for parameter estimation and model selection from experimental data in systems biology using approximate bayesian computation," *Nature Protocols*, vol. 9, no. 2, pp. 439–456, 2014. [Online]. Available : https://doi.org/10.1038/nprot.2014.025

[16] M. Secrier, T. Toni, and M. P. H. Stumpf, "The abc of reverse engineering biological signalling systems," *Mol. BioSyst.*, vol. 5, pp. 1925–1935, 2009. [Online]. Available : http://dx.doi.org/10.1039/B908951A

[17] G. Lillacci and M. Khammash, "Parameter estimation and model selection in computational biology," *PLOS Computational Biology*, vol. 6, no. 3, pp. 1–17, 03 2010. [Online]. Available : https://doi.org/10.1371/journal.pcbi.1000696

[18] H. Pacejka, E. Bakker, W. Pacejka, B. Hans, and M. Afsar, *Tyre and Vehicle Dynamics*. Elsevier Science, 2006.

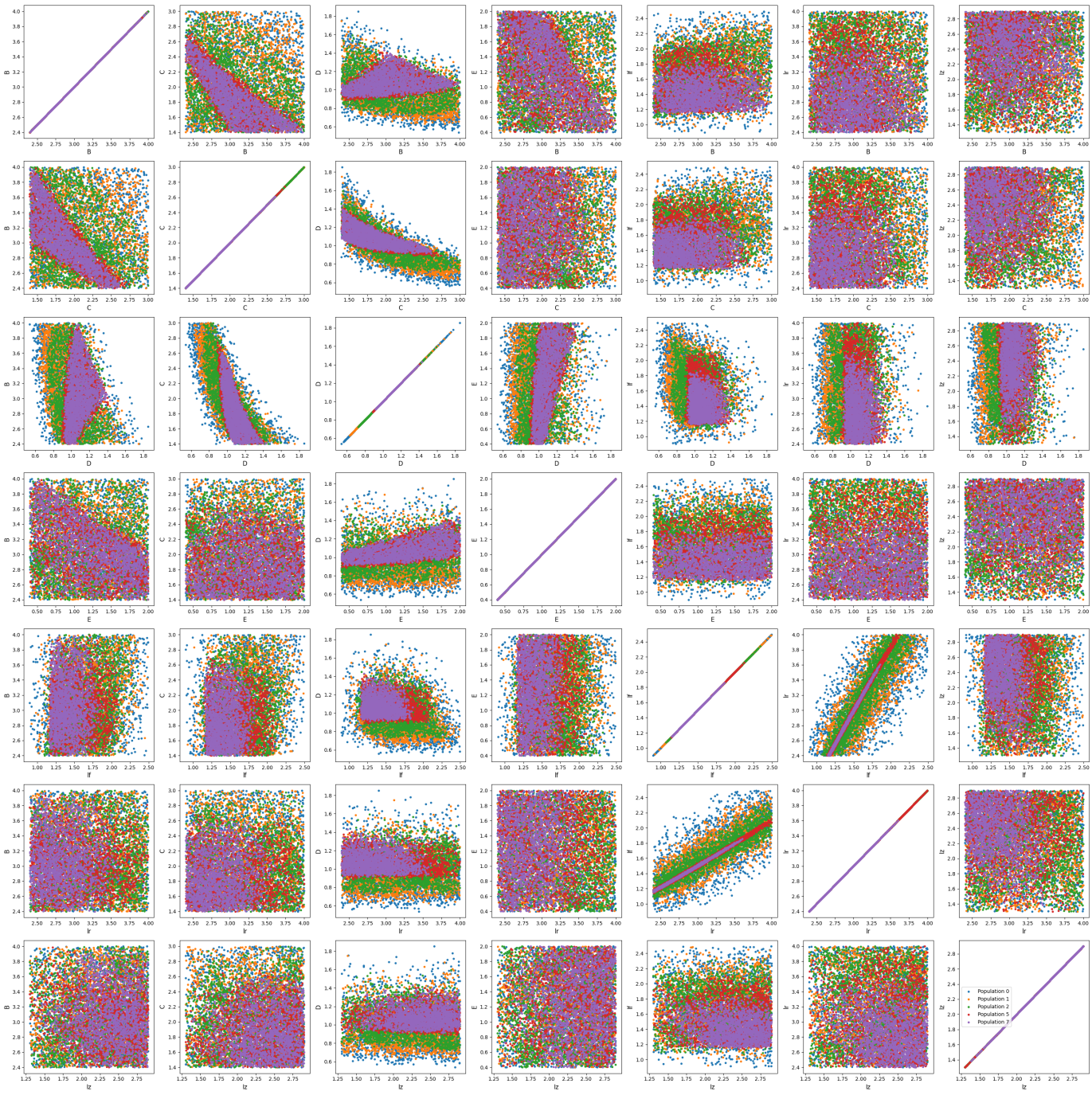


FIGURE 9 – Graphique de paires entre chacun des paramètres à estimer permettant de visualiser la convergence des échantillons vers la distribution de probabilité a posteriori au cours des itérations de l'algorithme *ABC-SMC*

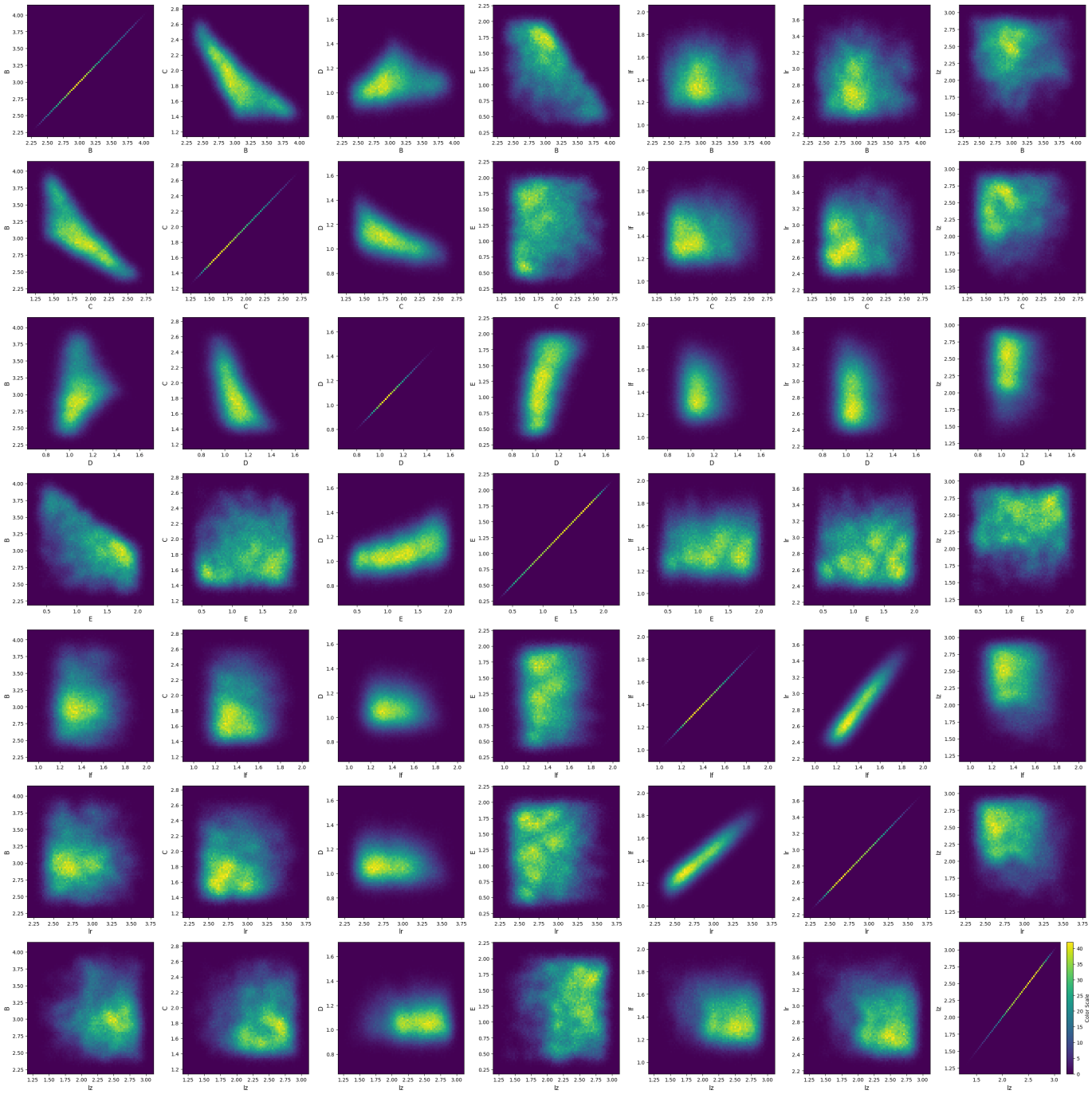


FIGURE 10 – Graphique de paires entre chacun des paramètres à estimer permettant de visualiser la densité de probabilité obtenue à partir de l'algorithme *ABC-SMC*. La densité est obtenue par estimation de densité par noyau appliqué aux échantillons correspondant à la dernière population de l'algorithme *ABC-SMC*

Application du chiffrement fonctionnel sur données confidentielles pour la conception de modèles d'apprentissage automatique

Tom Laborde
Naval Group
IRMAR - Université de Rennes 1
Toulon, France
tom.laborde@naval-group.com

Alexandre Gensse
Naval Group
Toulon, France
alexandre.gensse@naval-group.com

Philippe Chartier
Ravel Technologies
on leave from INRIA
Paris, France
philippe.chartier@inria.fr

Mohammed Lemou
Ravel Technologies
on leave from CNRS
Paris, France
mohammed.lemou@univ-rennes1.fr

Florian Méhats
Ravel Technologies
on leave from Univ Rennes
Paris, France
florian.mehats@univ-rennes1.fr

Fabien Chaillan
Naval Group
Toulon, France
fabien.chaillan@naval-group.com

Clément Gicquel
Naval Group
Toulon, France
clement.gicquel@naval-group.com

***Abstract*—L'accès aux données est un prérequis à la conception de modèle par apprentissage automatique. Dans certains secteurs d'application, comme la santé, le bancaire ou la défense, les données sont jugées confidentielles si bien que leur partage est particulièrement complexe, voire impossible. En réponse à ce verrou applicatif, nous nous intéressons dans ces travaux au chiffrement fonctionnel, technique de cryptographie qui permet l'accès sécurisé à fonction spécifique de données chiffrées. En nous appuyant sur un cas d'application concret de l'industrie de défense, nous montrons que le chiffrement fonctionnel est d'ores et déjà applicable à des données représentatives de la réalité industrielle et étudions son impact sur les performances de modèles obtenus par apprentissage automatique.**

***Index Terms*—Privacy Preserving Machine Learning, Cryptography, Functional Encryption, Inner Product Functional Encryption, Data Access, Machine Learning**

I. Introduction

A mesure que les infrastructures et algorithmes de traitement des données se développent, l'apprentissage automatique (Machine Learning ML), s'impose comme une approche algorithmique de plus en plus courante dans de nombreux domaines d'application. En général, la performance des modèles obtenus par ML découle directement de la quantité et de la représentativité des données d'apprentissage [1], [2]. En conséquence, la capacité de mettre à disposition de concepteurs spécialisés un jeu de données d'apprentissage pertinent est un prérequis fondamental au développement de l'intelligence artificielle (IA).

Dans certains domaines industriels, les données sont considérées comme confidentielles et leur mise à disposition est particulièrement complexe. En particulier, dans les industries de défense, de santé et bancaire, les cas d'usage de ML les plus critiques, souvent ceux avec le plus de valeur ajoutée, sont les plus difficiles à étudier et expérimenter de par la sensibilité des données impliquées. En palliatif à ce verrou applicatif d'une part, les concepteurs peuvent recourir à des méthodes pour compenser la frugalité de données réelles telles que l'apprentissage par transfert [50], [51], l'apprentissage frugal [52]–[54] et l'augmentation de données [55], [56]. En réponse à ce verrou applicatif d'autre part, de plus en plus d'efforts de recherche académique et industrielle sont menés dans le domaine du PPML (Privacy-Preserving Machine Learning) [3] et visent à intégrer des techniques de préservation de la confidentialité dans les architectures de traitement de données ou même directement au sein des algorithmes. Parmi les différentes approches existantes pour garantir la confidentialité des données, certaines reposent sur des techniques comme l'anonymisation [4]–[6], tandis que d'autres, plus récentes, exploitent des techniques de cryptographie sécurisées par conception [7]–[10]. Parmi ces dernières, on relève entre autres l'exploitation du chiffrement homomorphe (Homomorphic Encryption HE) et du chiffrement fonctionnel (Functional Encryption FE).

Le chiffrement homomorphe [11] est une technique permettant la réalisation d'opérations mathématiques sur des données préalablement chiffrées, sans accès aux données en clair. Dans ce contexte, seul le propriétaire des données a la capacité de déchiffrer le résultat. L'utilisation du HE dans les protocoles d'apprentissage automatique fait l'objet de recherches actives

[12]–[14], mais les progrès dans ce domaine peinent à suivre le rythme de l'augmentation constante des charges de calcul qu'impliquent l'apprentissage automatique.

Par contraste, le chiffrement fonctionnel permet la réalisation d'opérations sur données préalablement chiffrées avec obtention du résultat en clair ; seul le propriétaire des données décide des opérations applicables. Souvent considéré comme alternative au chiffrement homomorphe (HE) [15], [16], le chiffrement fonctionnel offre un panel sensiblement différent de cas d'applications industrielles : tout en garantissant la confidentialité du reste des données brutes, un tiers autorisé a la capacité d'extraire par lui-même une information choisie.

Les progrès récents en chiffrement fonctionnel semblent offrir de nouvelles perspectives d'application. Nous nous intéressons ici en particulier au schéma proposé par Agrawal *et al.* [17], revisité par Mera *et al.* [18], qui rend possible le calcul parallélisé de produits scalaires en s'appuyant sur l'extension Ring Learning With Errors (RLWE) du problème Learning With Errors (LWE).

Après un bref historique du chiffrement fonctionel (FE) et quelques définitions utiles, nous proposons dans ce travail une application du FE sur un cas d'usage du domaine du naval de défense. En nous appuyant sur un scénario de reconnaissance acoustique sous-marine, nous démontrons ainsi la maturité du FE face à des données réelles en amont d'une conception de modèle par apprentissage automatique profond et en étudions les divers impacts. Enfin, nous discutons de la pertinence du FE et des progrès techniques envisageables à court-terme.

II. Historique du chiffrement fonctionnel

Le chiffrement fonctionel (FE) est une méthode de chiffrement qui généralise le chiffrement basé sur l'identité (Identity-Based Encryption IBE) [19]–[23], le chiffrement basé sur l'attribut (Attribute-Based Encryption ABE) [24]–[30] et le chiffrement prédicatif (Predicate Encryption PE) [31]–[35]. Les premiers concepts de chiffrement fonctionnel sont introduits en 2010 par O'Neill [36] et en 2011 par Boneh, Sahai et Waters [37], qui le définissent comme un schéma permettant de déchiffrer le résultat $f(x)$ d'un calcul sélectif f d'un message x préalablement chiffré.

En 2015, Abdalla, Bourse, De Caro et Pointcheval [38] présentent le premier schéma IPFE (Inner Product Functional Encryption) efficace permettant de calculer le produit scalaire.

En 2016, Agrawal et al. [17] proposent des constructions basées sur les hypothèses standards DDH et LWE résistantes aux attaques adaptatives. Ils prouvent peu après que ces mêmes schémas sont également résistants aux attaques adaptatives SIM (AD-SIM) [39].

Les schémas IPFE s'appuient communément sur un processus en quatre étapes :

- SETUP : Génération des clés
- ENCRYPT : Chiffrement des données
- KEYGEN : Génération des clés fonctionnelles
- DECRYPT : Déchiffrement du produit scalaire.

Une définition plus détaillée de ce processus est donnée ci-après (Définition 2).

Bien que limitée aux opérations linéaires par leur restriction au produit scalaire, les schémas IPFE peuvent trouver des applications immédiates dans des domaines variés, impliquant des opérations comme entre autres le calcul de distance Manhattan, de moyenne pondérée ou de transformée de Fourier. De fait, des premiers résultats d'applications de schémas IPFE au ML ont été rapportés à la communauté par [40]–[43].

En 2022, Mera *et al.* [18] ont proposé un schéma IPFE basé sur le RLWE, exploitant les propriétés des anneaux polynomiaux quotientés afin de réduire la charge de calcul afférente au schéma et la taille des messages chiffrés. Nous exploitons ce schéma dans l'expérimentation rapportée au paragraphe IV.

III. Définitions

Définition 1: Soient les entiers $n, m \geq 1, p \geq 2$, un paramètre réel $\alpha \in [0, 1]$ et un vecteur secret $s \in \mathbb{Z}_p^n$. Pour $a \in \mathbb{Z}_p^n$ un vecteur tiré uniformément, et $e \hookleftarrow \mathcal{D}_{\alpha p}$ un bruit gaussien discret, on note $\mathcal{A}_{n,p,\alpha,s}$ la distribution du couple

$$(a,\ a \cdot s + e \mod p).$$

Le problème $\text{LWE}_{n,m,\alpha,p,s}$ [44]–[46], communément appelé problème de recherche LWE, consiste à retrouver s à partir de m tirages suivant $\mathcal{A}_{n,p,\alpha,s}$. Sa variante décisionnelle, communément appelée problème de décision LWE, consiste à distinguer $(\mathcal{A}_{n,p,\alpha,s})^m$ de la distribution uniforme $\mathcal{U}(\mathbb{Z}^{n+1})^m$. Un algorithme de décision calculable en temps polynomial (Probabilistic Polynomial-Time PPT) résout un problème difficile s'il atteint un avantage non négligeable au jeu de sécurité associé, ou en d'autres termes, s'il parvient avec une probabilité significativement supérieure à celle d'un oracle aléatoire à résoudre le problème.

Le problème RLWE [47], [48] est une extension du problème LWE. Soit R_p l'anneau polynomial $R_p = \mathbb{Z}_p[X]/(X^n + 1)$, avec n une puissance de 2 positive, et p un nombre premier tel que $p \equiv 1 \mod 2n$. Connaissant un couple $(a, as+e)$ avec a tiré uniformément dans R_p, $s \in R_p$ et e un bruit gaussien bien choisi [49], il n'y a pas d'algorithme PPT capable de retrouver s avec un avantage significatif.

Définition 2: Soient X un espace de messages, Y un espace de messages dual, E un espace de résultats et F une application définie sur $X \times Y \to E$. Un schéma de chiffrement fonctionnel pour F est un tuple d'algorithmes PPT (SETUP, ENCRYPT, KEYGEN, DECRYPT) tel que

- $\text{SETUP}(1^\lambda) \to (Mpk, Msk)$ prend en entrée le paramètre de sécurité λ et renvoie la clé publique Mpk et la clé secrète Msk.
- $\text{ENCRYPT}(Mpk, x) \to c_x$ prend en entrée la clé publique Mpk et un message $x \in X$ et renvoie un chiffré c_x du message.
- $\text{KEYGEN}(Msk, y) \to Sk_y$ prend en entrée la clé secrète Msk et un message dual $y \in Y$ et renvoie la clé fonctionnelle Sk_y associée.

- DECRYPT$(Sk_y, c_x) \rightarrow F(x, y)$ prend en entrée la clé fonctionnelle Sk_y et le chiffré c_x et renvoie le résultat $F(x, y) \in E$.

Dans le cas d'un schéma IPFE, la fonctionnalité $F(x, y)$ calculée lors du déchiffrement est le produit scalaire $\langle x, y \rangle$. Dans ce type de schéma, les espaces X et Y sont définis par $X = \{0, ..., B_x - 1\}^\ell$ et $Y = \{0, ..., B_y - 1\}^\ell$. Les paramètres $B_x, B_y \in \mathbb{N}^*$ sont les nombres de valeurs possibles pour chaque composante de respectivement les vecteurs de messages x et les vecteurs de messages duaux y. La valeur $\ell \in \mathbb{N}^*$ est la taille de ces vecteurs. Dans le cas des schémas IPFE basés sur le RLWE, il est possible de paralléliser le calcul de n chiffrements fonctionnels, n étant défini par l'anneau polynomial utilisé.

Définition 3: On rappelle que la transformée de Fourier S d'un signal discret $s \in [-1, 1]^N$ de $N \in \mathbb{N}^*$ échantillons est définie par

$$S = \left(\sum_{n=0}^{N-1} s(n) e^{-2i\pi kn/N} \right)_{k=0,\cdots,N-1}.$$

En définissant la matrice de Fourier $M_{\mathcal{F}} \in \mathbb{C}^{N \times N}$ telle que

$$M_{\mathcal{F}} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega^1 & \omega^2 & \dots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \dots & \omega^{(N-1)^2} \end{pmatrix}$$

avec $\omega = e^{-2i\pi/N}$, il est possible d'exprimer la transformée de Fourier discrète comme l'application linéaire

$$S = s^\top \times M_{\mathcal{F}}$$

Chaque ligne de $M_{\mathcal{F}}$ est associée à une composante de S, et donc à une bande de fréquence spécifique. Pour $0 \leq i < N$, la $i^{\text{ème}}$ ligne de $M_{\mathcal{F}}$ est ainsi associée à la fréquence $i \times \frac{F_e}{N}$, où F_e est la fréquence d'échantillonage de s.

IV. Application

A. Problématique et objectifs

Comme annoncé en introduction, nous nous intéressons à la problématique de la mise à disposition sécurisée de données jugées confidentielles à des concepteurs de modèles d'intelligence artificielle par ML. En particulier, nous considérons le cas des données de l'industrie de défense qui sont souvent considérées comme confidentielles.

L'accès aux données dans l'industrie de défense par un concepteur d'IA est par essence difficile, voire impossible. Pour développer les technologies IA dans ce secteur, les concepteurs s'appuient donc sur des techniques palliant le manque de données réelles comme l'apprentissage par transfert [50], [51], l'apprentissage frugal [52]–[54] ou l'augmentation de données [55], [56].

Dans l'hypothèse où l'information classifiée est identifiée, le FE peut être une réponse au verrou d'accès aux données de manière sécurisée. En effet, en définissant une fonction dont le résultat ne donne pas d'information sur l'information classifiée, il est acceptable de partager ses résultats. Notez bien qu'en pratique, cette hypothèse ne se vérifie que dans certains cas d'application.

En considérant cette hypothèse acquise, nous définissons le scénario d'application suivant : une marine souhaite faire concevoir un algorithme de reconnaissance automatique d'un type particulier d'émission sonar exploitant les données issues des enregistrements à la mer des sonars passifs de sa flotte. Par échanges avec un concepteur de modèle IA par ML elle identifie un jeu d'apprentissage pertinent, et décrète que l'information non-sensible strictement utile à ce cas d'usage est l'information sous-résolue à 10 bits contenue dans la bande de fréquence 450Hz à 600Hz.

La marine chiffre donc ses données et les envoie au concepteur de modèle. En parallèle, elle est en mesure d'envoyer les clés fonctionnelles permettant de déchiffrer les bandes de fréquences choisies. Notez bien que les données chiffrées ainsi le sont une fois pour toute : dans l'éventualité où d'autres cas d'usage sur ces données devraient être traités, il suffira alors de générer des clés fonctionnelles supplémentaires sans ré-itérer le chiffrement de l'intégralité des données archivées jusqu'alors. Remarquons également qu'au cours de ce processus, le concepteur à accès uniquement à la part des données que le propriétaire a choisi de partager.

Nous proposons ci-après les résultats de deux expérimentations. La première démontre la faisabilité et explicite le schéma de FE applicable. La seconde rapporte l'impact de l'utilisation de ce schéma FE sur les performances finales du modèle IA obtenu par ML.

B. Expérimentation 1 : Application du FE à un signal acoustique sous-marin

Nous considérons un signal acoustique sous-marin s, de durée $D = 10$ s et de fréquence d'échantillonnage $Fe = 22050$ Hz. Nous visons le déchiffrement fonctionnel de ses composantes fréquentielles strictement comprises entre 450 Hz et 600 Hz, telles qu'illustrées sur sa représentation en temps-fréquence par transformée de Fourier discrète à $N_{fft} = 1024$ points, et donc de résolution fréquentielle d'environ $21,5$Hz, avec recouvrement de fenêtres temporelles de 97,4%.

Nous exploitons le schéma de chiffrement fonctionnel proposé par Mera *et al.* [18], en définissant les paramètres du schéma de chiffrement en cohérence avec les spécifications du cas d'application :

- Résolution des vecteurs messages = 5 bits.
- Résolution des vecteurs messages duaux = 5 bits.
- Taille des vecteurs = 1024.
- Nombre de déchiffrements parallèles = 8192.

Le reste des paramètres est choisi de sorte à vérifier les conditions d'exactitude du déchiffrement fonctionnel. La sécurité

du schéma est estimée par l'outil *Lattice estimator*[1] à 246.2 bits. Pour rappel, le *Lattice Estimator* est un outil collaboratif mis en ligne par l'équipe de Martin R. Albrecht (King's College, Londres) régulièrement actualisé, qui permet notamment, à partir des principales attaques connues, d'évaluer la sécurité pratique des schémas basés sur les problèmes LWE, et RLWE par réduction. Notez cependant que la sécurité estimée n'a pas une valeur figée et qu'elle varie en fonction des avancées en cryptanalyse et de la puissance de calcul potentielle des attaquants.

Le signal acoustique s est porté à la résolution spécifiée puis chiffré en c_s en utilisant la clé publique. Sont ensuite définis 7 vecteurs fonctions $y_{21}, y_{22}, \ldots, y_{27}$, respectivement associés aux $21^{\text{eme}}, 22^{\text{eme}}, \ldots, 27^{\text{eme}}$ lignes de la matrice de Fourier $M_{\mathcal{F}}$. A partir de ces vecteurs fonctions et de la clé secrète sont générés les 7 couples de clés fonctionnelles $(Sk_{21re}, Sk_{21im}), (Sk_{22re}, Sk_{22im}), \ldots, (Sk_{27re}, Sk_{27im})$; les vecteurs fonctions étant complexes, chacun donne lieu à deux clés fonctionnelles. Le chiffré c_s et ces clés fonctionnelles sont ensuite utilisés pour obtenir par déchiffrements fonctionnels les composantes visées.

La Figure 1 représente le contenu spectrale du signal restreint à la bande fréquentielle 0-4000 Hz , tel qu'il aurait pu être calculé avec un accès total aux valeurs du signal s. La Figure 2 représente le résultat obtenu lors de cette expérimentation par déchiffrements fonctionnels : les composantes fréquentielles strictement comprises entre 450 Hz et 600 Hz. La quantification du signal est ramené à 10 bits de précision comme spécifié.

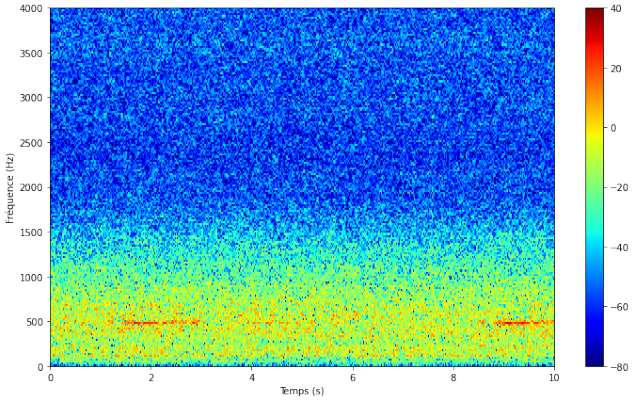


Fig. 1. Représentation en temps-fréquence par transformée de Fourier du signal en clair, sur la bande utile 0-4000Hz, avec $N_{fft} = 1024$ et $F_e = 22050$Hz. En abscisse, le temps en secondes. En ordonnée, la fréquence en Hertz. Les valeurs sont représentées sur l'échelle de couleur à droite en décibels.

Il est important de remarquer que le calcul de la transformation de Fourier nécessaire à l'obtention des composantes ciblées est réalisé lors du déchiffrement fonctionnel : le message chiffré est le signal acoustique. Pour rappel, les temps

[1]https://github.com/malb/lattice-estimator

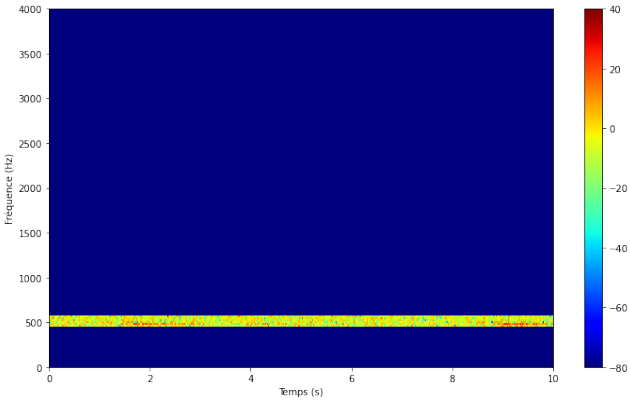


Fig. 2. Représentation en temps-fréquence obtenus par 14 déchiffrements fonctionnels. En abscisse, le temps en secondes. En ordonnée, la fréquence en Hertz. Les valeurs sont représentées sur l'échelle de couleur à droite en décibels.

de calcul afférents à chaque étape du chiffrement fonctionnel (Cf. 2) rapportés par Mera *et al.* [18] sont de

$$\begin{cases} SETUP = 1743 \text{ ms}, \\ ENCRYPT = 1388 \text{ ms}, \\ KEYGEN = 70 \text{ ms}, \\ DECRYPT = 45 \text{ ms}, \end{cases}$$

pour un processeur Intel i9-9880H fonctionnant à une fréquence maximale de 4,8 GHz.

C. Expérimentation 2 : Évaluation de l'impact du chiffrement fonctionnel sur les performances finales du modèle d'intelligence artificielle obtenu par apprentissage automatique

En exploitant le processus de déchiffrement fonctionnel décrit dans l'expérimentation 1 (Paragraphe IV-B), nous cherchons à évaluer l'impact de l'utilisation du chiffrement fonctionnel et de la restriction de l'information à disposition en apprentissage sur les performances d'un modèle IA de reconnaissance acoustique sous-marine (ASM). Nous proposons dans ce paragraphe une expérimentation s'appuyant sur le scénario de reconnaissance d'émission SONAR présenté au paragraphe IV-A qui émule la conception de modèles basés sur apprentissage.

Nous exploitons dans cette expérience un jeu de données ASM constitué de données réelles mises à disposition par l'université de San Francisco [58]. Nous décomposons aléatoirement ce jeu de données en un jeu d'apprentissage de 602 exemples, dont 78 exemples positifs annotés "ES", et un jeu de test de 147 exemples, dont 19 positifs. En fonction des traitements appliqués, nous distinguons trois jeux de données :

- Jeu de données A : Restriction de la bande passante à $0 - 4000$ Hz. Ce jeu de données est considéré comme référence puisque représentatif des données inaltérées par chiffrement fonctionnel ou sélection de fréquence.

TABLE I
MESURES DE PERFORMANCES DES MODÈLES 1, 2 ET 3 SUR LES JEUX DE DONNÉES A, B ET C.
MOYENNES ET ÉCART-TYPES SUR 20 APPRENTISSAGES PAR CONFIGURATION.

Modèle	Jeu de données	Vrais Positifs	Vrais Négatifs	Faux positifs	Faux négatifs	AUC
1	A	16.7 ($\pm$1.2)	120.4 ($\pm$1.7)	7.6 ($\pm$1.7)	2.3 ($\pm$1.2)	0.962 ($\pm$0.025)
1	B	15.3 ($\pm$1.6)	120.5 ($\pm$1.6)	7.6 ($\pm$1.6)	3.7 ($\pm$1.6)	0.906 ($\pm$0.055)
1	C	14.5 ($\pm$1.9)	120.7 ($\pm$2.3)	7.3 ($\pm$2.3)	4.5 ($\pm$1.9)	0.814 ($\pm$0.049)
2	A	11.7 ($\pm$2.4)	121.8 ($\pm$2.1)	6.3 ($\pm$1.6)	7.3 ($\pm$2.4)	0.948 ($\pm$0.025)
2	B	10.2 ($\pm$0.9)	122.6 ($\pm$1.7)	5.4 ($\pm$1.7)	8.9 ($\pm$0.9)	0.844 ($\pm$0.045)
2	C	9.6 ($\pm$1.0)	121.0 ($\pm$6.3)	7.1 ($\pm$6.3)	9.4 ($\pm$1.0)	0.817 ($\pm$0.056)
3	A	14.5 ($\pm$1.9)	120.7 ($\pm$2.3)	7.3 ($\pm$2.3)	4.5 ($\pm$1.9)	0.937 ($\pm$0.027)
3	B	10.5 ($\pm$1.1)	122.5 ($\pm$0.8)	5.5 ($\pm$0.8)	8.6 ($\pm$1.1)	0.841 ($\pm$0.045)
3	C	9.8 ($\pm$1.2)	121.7 ($\pm$2.5)	6.4 ($\pm$2.5)	9.3 ($\pm$1.2)	0.843 ($\pm$0.030)

TABLE II
IMPACTS DE LA RÉDUCTION DE RÉSOLUTION
MESURÉS SUR LE JEU DE DONNÉES B EN POURCENTAGE DE LA PERFORMANCE DE RÉFÉRENCE OBTENUE SUR LE JEU DE DONNÉES A.

Modèle	Vrais Positifs	Vrais Négatifs	Faux positifs	Faux négatifs	AUC
1	-8.4	$+0.1$	0.0	$+60.9$	-5.8
2	-12.8	$+0.7$	-14.3	$+21.9$	-11.0
3	-27.6	$+1.5$	-24.7	$+91.1$	-10.2

TABLE III
IMPACTS DE LA RÉDUCTION DE RÉSOLUTION ET DE LA RESTRICTION D'INFORMATIONS
MESURÉS SUR LE JEU DE DONNÉES C EN POURCENTAGE DE LA PERFORMANCE DE RÉFÉRENCE OBTENUE SUR LE JEU DE DONNÉES A.

Modèle	Vrais Positifs	Vrais Négatifs	Faux positifs	Faux négatifs	AUC
1	-13.2	$+0.2$	-3.9	$+95.7$	-15.4
2	-17.9	$+0.7$	$+12.7$	$+28.8$	-13.8
3	-32.4	$+0.8$	-12.3	$+106.7$	-10.0

- Jeu de données B : Restriction de la bande passante à $0-4000$ Hz et réduction de la résolution par application du schéma de chiffrement fonctionnel tel que rapporté au paragraphe IV-B. Ce jeu de données est utilisé pour mesurer l'impact de la réduction de résolution.
- Jeu de données C : Restriction de la bande passante à $450-600$ Hz et réduction de la résolution par application du schéma de chiffrement fonctionnel tel que rapporté au paragraphe IV-B. Ce jeu de données est utilisé pour mesurer l'impact simultané de la réduction de résolution et de la restriction d'information à disposition en phase d'apprentissage.

Sur la base des travaux rapportés par Artusi et Chaillan [50], nous définissons trois modèles d'apprentissage de tailles différentes :

- Modèle 1 : 284690 paramètres
- Modèle 2 : 39090 paramètres
- Modèle 3 : 16498 paramètres

Ces modèles s'appuient sur des neurones à filtres de convolutions (CNN) et des LSTM (Long Short-Term Memory) pour proposer une classification binaire distinguant les exemples positifs, annotés "ES", des exemples négatifs. Le CNN (Convolutional Neural Networks en anglais), ou réseau de neurones convolutifs, est un type de réseaux de neurones souvent utilisés pour analyser des données représentées en images [59]. Le LSTM (Long Short-Term Memory en anglais), est une variante des réseaux de neurones récurrents (RNN) conçue pour apprendre des dépendances à court et long terme et prendre en compte l'aspect séquentiel des données représentée en série temporelle [60].

Dans la volonté de minimiser le biais d'estimation des impacts sur les performances, nous avons choisi arbitrairement, en amont des expérimentations, des paramètres d'apprentissage communs aux trois modèles. Afin d'éviter de probables sur-apprentissages, nous utilisons les mécanismes de pondération de classes (Class Balancing), de dilution (Drop Out) et d'arrêt anticipé (Early Stopping). En réponse aux différences de taille de modèles, nous utilisons également un planificateur de taux d'apprentissage (Learning Rate Scheduler).

La table I rapporte le nombre de vrais positifs, vrais négatifs, faux positifs et faux négatifs mesurés par évaluation sur jeux de test respectifs des jeux de données A, B et C pour chaque modèle. Sont également rapportés les aires sous la courbe (AUC) de caractéristique opérationnelle de récepteur (COR). Les performances rapportées sont les moyennes et écart-types de chacune de ces métriques sur un ensemble de 20 apprentissages par modèle et par jeu de données.

La table II rapporte les impacts de la baisse de résolution induite par application du schéma de chiffrement fonctionnel. La table III rapporte les impacts de la baisse de résolution conjuguée à la restriction des informations disponibles à l'apprentissage.

Conformément à nos attentes, nous constatons les impacts successifs de la réduction de résolution et de la restriction des informations disponibles. Nous observons que la taille du

modèle influe sur ces impacts respectifs : plus le modèle est complexe, moins il est sensible à la réduction de résolution, mais plus il est sensible à la restriction d'informations disponible à l'apprentissage. En contraste avec notre approche limitant le biais, il semble qu'adapter les architectures de modèles IA et les paramètres de chiffrements en fonction du cas d'usage est une nécessité pour minimiser la perte de performance des modèles.

V. Conclusion

Nous avons abordé la question de l'accès aux données confidentielles, en particulier dans l'industrie de défense. En ce sens, nous avons rapporté des résultats d'expérimentations qui démontrent la faisabilité de l'application de l'IPFE à des données réelles et nous avons, sur un cas d'application spécifique et sans adaptations particulières, mesuré son impact sur les performances d'un modèle IA. Nous soulignons que l'approche proposée repose sur l'hypothèse que la fonction calculée ne délivre pas d'information sensible. Dans de futurs travaux nous prévoyons d'étudier les méthodes permettant de minimiser ces impacts, que ce soit par des techniques relatives au protocole d'apprentissage ou par l'élargissement du panel de fonctions déchiffrables par les schémas de chiffrement fonctionnels : au-delà de l'IPFE, des schémas permettant le déchiffrement fonctionnel d'opérations quadratiques ont été récemment proposés [61], [62], voire expérimentés en laboratoire [63], [64]. Nous prévoyons de rapporter nos expérimentations de ces schémas sur données industrielles.

References

[1] HASTIE, Trevor, TIBSHIRANI, Robert, FRIEDMAN, Jerome H., *et al. The elements of statistical learning: data mining, inference, and prediction*. New York : springer, 2009.

[2] IAN, H. Witten et EIBE, Frank. Data Mining: Practical machine learning tools and techniques. 2005.

[3] Xu, R. (2020). Functional encryption based approaches for practical privacy-preserving machine learning (Doctoral dissertation, University of Pittsburgh).

[4] Latanya Sweeney. k-anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10(05):557–570, 2002.

[5] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. ACM Transactions on Knowledge Discovery from Data (TKDD), 1(1):3–es, 2007.

[6] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In 2007 IEEE 23rd International Conference on Data Engineering, pages 106–115. IEEE, 2007.

[7] Gilbert Wondracek, Thorsten Holz, Engin Kirda, and Christopher Kruegel. A practical attack to deanonymize social network users. In 2010 IEEE Symposium on Security and Privacy, pages 223–238. IEEE, 2010.

[8] Md Atiqur Rahman, Tanzila Rahman, Robert Laganière, Noman Mohammed, and Yang Wang. Membership inference attack against differentially private deep learning model. Transactions on Data Privacy, 11(1):61–79, 2018.

[9] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In 2017 IEEE Symposium on Security and Privacy (SP), pages 3–18. IEEE, 2017.

[10] Jianwei Qian, Xiang-Yang Li, Chunhong Zhang, and Linlin Chen. De-anonymizing social networks and inferring private attributes using knowledge graphs. In IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, pages 1–9. IEEE, 2016.

[11] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Proceedings of the fortyfirst annual ACM symposium on Theory of computing (2009), pp. 169–178.

[12] Y. Aono, T. Hayashi, L. Wang, and S. Moriai (2017). Privacy-preserving deep learning via additively homomorphic encryption. IEEE Transactions on Information Forensics and Security, 13(5), 1333-1345.

[13] H. Fang and Q. Qian (2021). Privacy preserving machine learning with homomorphic encryption and federated learning. Future Internet, 13(4), 94.

[14] A. Falcetta and M. Roveri (2022). Privacy-preserving deep learning with homomorphic encryption: An introduction. IEEE Computational Intelligence Magazine, 17(3), 14-25.

[15] J. Alwen, M. Barbosa, P. Farshim, R. Gennaro, S. D. Gordon, S. Tessaro, and D. A. Wilson. On the relationship between functional encryption, obfuscation, and fully homomorphic encryption. In IMA International Conference on Cryptography and Coding (2013), Springer, pp. 65–84.

[16] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. SIAM Journal on Computing 45, 3 (2016), 882–929.

[17] S. Agrawal, B. Libert, and D. Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In Annual International Cryptology Con ference (2016), Springer, pp. 333–362.

[18] J. M. B. Mera, A. Karmakar, T. Marc, A. Soleimanian (2022, February). Efficient lattice-based inner-product functional encryption. In Public-Key Cryptography–PKC 2022: 25th IACR International Conference on Practice and Theory of Public-Key Cryptography, Virtual Event, March 8–11, 2022, Proceedings, Part II (pp. 163-193). Cham: Springer International Publishing.

[19] Dan Boneh and Matthew K. Franklin. "Identity-Based Encryption from the Weil Pairing". In: CRYPTO 2001. Ed. by Joe Kilian. Vol. 2139. LNCS. Springer, Heidelberg, Aug. 2001, pp. 213–229 (cit. on p. 2).

[20] Dan Boneh and Xavier Boyen. "Secure Identity Based Encryption Without Random Oracles". In: CRYPTO 2004. Ed. by Matthew Franklin. Vol. 3152. LNCS. Springer, Heidelberg, Aug. 2004, pp. 443–459 (cit. on p. 2).

[21] Brent Waters. "Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions". In: CRYPTO 2009. Ed. by Shai Halevi. Vol. 5677. LNCS. Springer, Heidelberg, Aug. 2009, pp. 619–636 (cit. on p. 2).

[22] Shweta Agrawal, Dan Boneh, and Xavier Boyen. "Efficient Lattice (H)IBE in the Standard Model". In: EUROCRYPT 2010. Ed. by Henri Gilbert. Vol. 6110. LNCS. Springer, Heidelberg, May 2010, pp. 553–572 (cit. on pp. 2, 108).

[23] Shweta Agrawal, Dan Boneh, and Xavier Boyen. "Lattice Basis Delegation in Fixed Dimension and Shorter-Ciphertext Hierarchical IBE". In: CRYPTO 2010. Ed. by Tal Rabin. Vol. 6223. LNCS. Springer, Heidelberg, Aug. 2010, pp. 98–115 (cit. on p. 2).

[24] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data". In: ACM CCS 06. Ed. by Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati. Available as Cryptology ePrint Archive Report 2006/309. ACM Press, Oct. 2006, pp. 89–98 (cit. on p. 2).

[25] Rafail Ostrovsky, Amit Sahai, and Brent Waters. "Attribute-based encryption with non-monotonic access structures". In: ACM CCS 07. Ed. by Peng Ning, Sabrina De Capitani di Vimercati, and Paul F. Syverson. ACM Press, Oct. 2007, pp. 195–203 (cit. on p. 2).

[26] Vipul Goyal, Abhishek Jain, Omkant Pandey, and Amit Sahai. "Bounded Ciphertext Policy Attribute Based Encryption". In: ICALP 2008, Part II. Ed. by Luca Aceto, Ivan Damgård, Leslie Ann Goldberg, Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz. Vol. 5126. LNCS. Springer, Heidelberg, July 2008, pp. 579–591 (cit. on p. 2).

[27] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. "Fully Secure Functional Encryption: Attribute-Based Encryption and (Hierarchical) Inner Product Encryption". In: EUROCRYPT 2010. Ed. by Henri Gilbert. Vol. 6110. LNCS. Springer, Heidelberg, May 2010, pp. 62–91 (cit. on p. 2).

[28] Brent Waters. "Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization". In: PKC 2011. Ed. by Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi. Vol. 6571. LNCS. Springer, Heidelberg, Mar. 2011, pp. 53–70 (cit. on p. 2).

[29] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. "Attribute-based encryption for circuits". In: 45th ACM STOC. Ed. by Dan Boneh,

Tim Roughgarden, and Joan Feigenbaum. ACM Press, June 2013, pp. 545–554 (cit. on p. 2).

[30] Zvika Brakerski and Vinod Vaikuntanathan. “Circuit-ABE from LWE: Unbounded Attributes and Semi-adaptive Security”. In: CRYPTO 2016, Part III. Ed. by Matthew Robshaw and Jonathan Katz. Vol. 9816. LNCS. Springer, Heidelberg, Aug. 2016, pp. 363–384. doi: 10.1007/978-3-662-53015-3_13 (cit. on p. 2).

[31] Jonathan Katz, Amit Sahai, and Brent Waters. “Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products”. In: EUROCRYPT 2008. Ed. by Nigel P. Smart. Vol. 4965. LNCS. Springer, Heidelberg, Apr. 2008, pp. 146–162 (cit. on p. 2).

[32] Jonathan Katz and Arkady Yerukhimovich. “On Black-Box Constructions of Predicate Encryption from Trapdoor Permutations”. In: ASIACRYPT 2009. Ed. by Mitsuru Matsui. Vol. 5912. LNCS. Springer, Heidelberg, Dec. 2009, pp. 197–213 (cit. on p. 2).

[33] Tatsuaki Okamoto and Katsuyuki Takashima. “Hierarchical Predicate Encryption for Inner-Products”. In: ASIACRYPT 2009. Ed. by Mitsuru Matsui. Vol. 5912. LNCS. Springer, Heidelberg, Dec. 2009, pp. 214–231 (cit. on pp. 2, 37).

[34] Romain Gay, Pierrick Méaux, and Hoeteck Wee. “Predicate Encryption for Multi-dimensional Range Queries from Lattices”. In: PKC 2015. Ed. By Jonathan Katz. Vol. 9020. LNCS. Springer, Heidelberg, Mar. 2015, pp. 752–776. doi: 10.1007/978-3-662-46447-2_34 (cit. on p. 2).

[35] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. “Predicate Encryption for Circuits from LWE”. In: CRYPTO 2015, Part II. Ed. by Rosario Gennaro and Matthew J. B. Robshaw. Vol. 9216. LNCS. Springer, Heidelberg, Aug. 2015, pp. 503–523. doi: 10.1007/978-3-662-48000-7_25 (cit. on p. 2).

[36] A. O’Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2009/556, 2010. https://eprint.iacr.org/2010/556.

[37] D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In Theory of Cryptography Conference (2011), Springer, pp. 253–273.

[38] M. Abdalla, F. Bourse, A. De Caro, and D. Pointcheval. Simple functional encryption schemes for inner products. In IACR International Workshop on Public Key Cryptography (2015), Springer, pp. 733–751.

[39] Agrawal, S., Libert, B., Maitra, M., and Titiu, R. Adaptive simulation security for inner product functional encryption. In IACR International Conference on Public-Key Cryptography (2020), Springer, pp. 34–64.

[40] T. Marc, M. Stopar, J. Hartman, M. Bizjak, J. Modic (2019). Privacy-enhanced machine learning with functional encryption. In Computer Security–ESORICS 2019: 24th European Symposium on Research in Computer Security, Luxembourg, September 23–27, 2019, Proceedings, Part I 24 (pp. 3-21). Springer International Publishing.

[41] T. Ryffel, E. Dufour-Sans, R. Gay, F. Bach and D. Pointcheval (2019). Partially encrypted machine learning using functional encryption. arXiv preprint arXiv:1905.10214.

[42] T. Ryffel, D. Pointcheval, F. Bach, E. Dufour-Sans and R. Gay (2019). Partially encrypted deep learning using functional encryption. Advances in Neural Information Processing Systems, 32.

[43] E. Dufour-Sans, R. Gay and D. Pointcheval (2018). Reading in the dark: Classifying encrypted digits with functional encryption. Cryptology ePrint Archive.

[44] REGEV, Oded. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, vol. 56, no 6, p. 1-40, 2009.

[45] Peikert, Chris. ”Public-key cryptosystems from the worst-case shortest vector problem.” *Proceedings of the forty-first annual ACM symposium on Theory of computing*. 2009.

[46] Lyubashevsky, Vadim, Chris Peikert, and Oded Regev. ”On ideal lattices and learning with errors over rings.” *Advances in Cryptology–EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30–June 3, 2010. Proceedings 29*. Springer Berlin Heidelberg, 2010.

[47] LU, Xianhui, LIU, Yamin, ZHANG, Zhenfei, *et al.* LAC: Practical ring-LWE based public-key encryption with byte-level modulus. *Cryptology ePrint Archive*, 2018.

[48] LYUBASHEVSKY, Vadim, PEIKERT, Chris, et REGEV, Oded. A toolkit for ring-LWE cryptography. In : *Advances in Cryptology–EUROCRYPT 2013: 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings 32*. Springer Berlin Heidelberg, 2013. p. 35-54.

[49] Vadim Lyubashevsky, Chris Peikert and Oded Regev. On ideal lattices and learning with errors over rings *in Annual international conference on the theory and applications of cryptographic techniques*, pages 1-23, 2010.

[50] E. Artusi and F. Chaillan (2019). Automatic recognition of underwater acoustic signature for naval applications. In 1st Maritime Situational Awareness Workshop MSAW 2019.

[51] Torrey, Shavlik : Transfer learning. In Handbook of research on machine learning applications and trends: algorithms, methods, and techniques (pp. 242-264). IGI global. 2010.

[52] Evchenko, Vanschoren, Hoos, Schoenauer, Sebag : Frugal machine learning. arXiv:2111.03731, 2021

[53] Wang, Yao, Kwok, Ni : Generalizing from a few examples: A survey on few-shot learning. ACM computing surveys (csur), 53(3), 1-34, 2020.

[54] Lake, Salakhutdinov, Gross, Tenenbaum : One shot learning of simple visual concepts. In Proceedings of the annual meeting of the cognitive science society (Vol. 33, No. 33), 2011.

[55] Shorten, Khoshgoftaar : A survey on image data augmentation for deep learning. Journal of big data, 6(1), 1-48. 2019.

[56] Van Dyk, Meng : The art of data augmentation. Journal of Computational and Graphical Statistics, 10(1), 1-50. 2001.

[57] BARKER, Elaine, BARKER, William, BURR, William, *et al.* NIST special publication 800-57. *NIST Special publication*, vol. 800, no 57, p. 1-142. 2007.

[58] https://maritime.org/sound/

[59] LECUN, Yann, BOTTOU, Léon, BENGIO, Yoshua, *et al.* Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, vol. 86, no 11, p. 2278-2324, 1998.

[60] HOCHREITER, Sepp et SCHMIDHUBER, Jürgen. Long short-term memory. *Neural computation*, vol. 9, no 8, p. 1735-1780. 1997.

[61] Carmen Baltico, Zaira Elisabetta, Dario Catalano Dario, Dario Fiore, Romain Gay. Practical functional encryption for quadratic functions with applications to predicate encryption. In Annual International Cryptology Conference, pages 67-98, 2017.

[62] Shweta Agrawal, Alon Rosen. Functional encryption for bounded collusions. In Theory of Cryptography Conference, pages 173-205, 2016.

[63] Edouard Dufour-Sans, Romain Gay and David Pointcheval. Reading in the dark: Classifying encrypted digits with functional encryption In Cryptology ePrint Archive, 2018.

[64] Xu, Runhua, James BD Joshi, Chao Li. Cryptonn: Training neural networks over encrypted data. In IEEE 39th International Conference on Distributed Computing Systems (ICDCS), pages 1199-1209, 2019.

Towards a Definition of Cognitive Warfare

MORELLE Marie
Univ. Bordeaux, Bdx INP – ENSC – IMS, THALES LAS France
33400 Talence, FRANCE
mmorelle@ensc.fr

MARION Damien
THALES LAS France, HEAL
33400 Talence, FRANCE
damien.marion@thalesgroup.com

CEGARRA Julien
INU Champollion
81012 Albi Cedex, FRANCE
julien.cegarra@univ-jfc.fr

ANDRÉ Jean-Marc
Univ. Bordeaux, Bdx INP – ENSC – IMS
33400 Talence, FRANCE
jean-marc.andre@ensc.fr

***Abstract*—Cognitive warfare is an emerging concept in the literature, linked to the development of new technologies and of knowledge on cognition, as well as the involvement of public opinion in conflicts. It is currently the subject of debate within NATO. Used to destabilize adversaries or make them "destroy themselves from the inside", it is becoming a major concern, and we need to learn how to detect it and protect ourselves against it. This paper aims to define cognitive warfare, other close concepts and its players, and to outline the issues at stake.**

Keywords—cognitive warfare, influence strategies, destabilization actions, intelligence

I. Introduction

This paper will provide some insights into cognitive warfare, its definition and related concepts. Bernal et al. [3] suggest that modern cognitive warfare emerged during the Cold War: to avoid another destructive open war between superpowers, they set up proxy conflicts, supporting and opposing small countries or armed groups against each other. In this context, numerous actions have been carried out discreetly, notably by the CIA, the FBI and the KGB. Since the 2000s, there has been an increase in destabilization actions [231], particularly by Russia in its attempts to influence elections, propaganda and cyberattacks, notably against the Baltic states, France and the United States, as described by Backes and Swab [1]. Du Cluzel explains that some countries, like Russia and China, are conducting research into neuroscience and technology for medical, social and military purposes, which could also serve as a means of action for cognitive warfare [9]. But it is in the field of economics that the term "cognitive warfare" was first used, to describe the influences and destabilizing actions implemented by companies, towards consumers, legislators (lobbying) or even competing companies [11].

Cognitive warfare thus represents a new mode of conflict blurring the boundaries between war and peace [21], not open warfare on a battlefield, but undeclared warfare aimed at influencing the cognitive mechanisms, notably the decision-making processes, of an adversary or competitor. It's a recent term, and not yet universally accepted. We propose a definition, and discuss the challenges of this new type of confrontation.

II. Cognitive Warfare

Cognitive warfare is an emerging concept aiming to weaken the enemy in order to gain a tactical or strategic advantage [18], in the military, economics, gaming, sports and other fields. It encompasses various operations carried out against the human mind, targeting both individual and collective cognition and decision-making. It can be carried out and suffered at different scales, and remotely: populations, soldiers, experts, engineers, technicians, groups or minorities of opinion, ethnic or religious, companies, communities, decision-makers, political, economic, religious, academic or military leaders [3, 7]…

It relies on NBIC (Nanotechnologies, Biotechnologies, Information technologies and Cognitive sciences) tools, such as digital tools and social networks, chemical substances, illusions, attention saturation [7], or exploitable cognitive biases of targets [22]. The objectives of cognitive warfare can be to conquer a territory, disrupt public services, bring about a change of government, influence elections, undermine confidence, inhibit critical thinking [9], or destabilize and influence a target population [3] by radicalizing opinion, discrediting governing bodies, triggering or inhibiting actions, etc. These actions are difficult to detect, as both targets and relays are unaware that they are victims.

Here are different examples of actions that could be considered as cognitive warfare.

1. Cognitive warfare launched via cyber tools: On the 23rd of February, 2022, just before the Russian offensive against Ukraine, almost 500 cyber-attacks were detected on Ukrainian networks. They targeted essentially the government, critical networks and energy suppliers. According to Malin [17], the goal was seemingly to prevent the Ukrainian government from communicating and providing vital resources to its population, so that they lose confidence in the government. Even though this action is widely considered as cyberwarfare, it can be studied in the perspective of cognitive warfare because it is part of a strategy using different tools with a broader goal of destabilization.
2. Cognitive warfare launched contacting directly key individuals: On the 6th of February 2023 in France, several women deputies from the Rassemblement National received a vocal message telling them one of their children was hospitalized, so that they would leave the National Assembly before a critical vote [24]. This was an intimidation attack trying to prevent the targeted people from voting and thus taking their role in a decision.
3. Cognitive warfare launched on social media: Our last example takes place in Mali, in April 2022. The Wagner mercenary group staged a human mass grave

and edited a video to make it look like the French military created it [14]. The French military managed to film them preparing this staged scene, allowing them to create a video refuting these accusations. According to Jousset and Bolchakova [14], the Wagner group would have used this opportunity to create an influence campaign on social networks to delegitimize the French army in Mali and be welcomed as liberators against French oppression.

4. Another example often used to describe cognitive warfare is the Havana Syndrome [26], which could potentially be a cognitive warfare attack launched using targeted nano or biological tools to induce fear for the key individuals who were targeted and cloud their judgment, but there is no definite cause officially identified, so this example is not supported by evidence to present day.

These examples show how cognitive warfare strategies can be led using different means and channels of action. We propose 3 criteria to help determine whether an action falls within the scope of cognitive warfare:

1. Its purpose is broader than what is immediately apparent: for example, a cyberattack launched in order to steal credit cards and use them to gain a monetary advantage would not be cognitive warfare; but if the aim is to instill doubt in customers' minds about the bank's ability to be secure, and the perpetrator communicates about this to discredit it, this could be an example of cognitive warfare.

2. Its nature is diffuse: it is often part of a strategy encompassing different actions and different means with a goal of chain reactions which can be difficult to measure in time and space. For example, an action could aim at changing behaviors in a population, so another group of people would have a different point of view on this population and then it would have an impact on some votes or decision-makers, and we can imagine a whole chain of resulting consequences and actions; it would thus prove difficult to find the origin of the attack and determine how long-term its effects can be.

3. Its target is cognition: cognitive warfare targets the human mind, representations and cognition, it aims at sowing doubt, preventing or influencing decisions, undermining the opponent's will [21], manipulating the way people see and interpret specific parts of the world around them [3]… In short, how people think and react [25].

III. Close Concepts

There are other concepts closely related to cognitive warfare. We propose a clarification of terminology to better organize and define related terms. Most of them are intertwined with cognitive warfare and/or can be used as tools to lead a cognitive warfare strategy, under the circumstances explained earlier.

A. *Information Warfare, or Information Operations*

NATO[1] describes information warfare as "an operation conducted to gain an informational advantage over the opponent". It focuses on information, its manipulation, its flow, the way it is protected or stolen, and the way it is used. Du Cluzel [20] reminds us that cognitive warfare, on the contrary, is "an action against the way we think, the way we process information and turn it into knowledge". Bernal et al. [3] argue that cognitive warfare is the "fight to control or alter the way people react to information". Intelligence, on the other hand, is a process of knowledge construction [5]. Unlike information warfare, cognitive warfare does not focus on tactical battlefield information, but also acts on information for the general public [3].

B. *Psychological Operations (PsyOps)*

According to Bernal et al. [3], U.S.-led PsyOps involve informational products that are either identifiable as officially U.S.-produced (white products), ambiguously sourced (gray products), or created to "seem as if they originate from a hostile source" (black products). They often have a military purpose. Bernal et al. point out that cognitive warfare works mostly with gray products, which can be denied, and that it "tends to target civilian social infrastructure and governments".

C. *Propaganda*

Propaganda is the transmission of communications, information, and messages for the purpose of causing changes in the consciousness or subconsciousness of the target population, in order to change attitudes and behaviors [4]. Its author assumes authorship, so it is directly attributable. Cognitive warfare is subtler than propaganda. It allows one to influence targets without them being aware of it, and to use these same targets as weapons to reach others. According to du Cluzel [9], with cognitive warfare, "everyone participates, mostly inadvertently, to information processing and knowledge formation in an unprecedented way". The author states that cognitive warfare "feeds on the techniques of disinformation and propaganda aimed at psychologically exhausting the receptors of information": they can therefore be cognitive warfare tools.

D. *Cyber Warfare*

Bernal et al. [3] define cyber warfare as "the use of cyberattacks with the intention of causing harm to a nation's assets". In our connected societies, especially with the Internet of Things, many functions are digitally controlled: "from construction equipment, to financial institutions, to civilian infrastructure, and even to military installations". Thus, cyberattacks can cause "massive damages not just in terms of time and data loss but in physical damage that can be measured in dollars and lives". Computer warfare falls under the technical domain, and the cognitive effect is a consequence, whereas for cognitive warfare, it is the goal of the action [7].

E. *Cyberpsychology*

According to Claverie and Kowalczuk [8], cyberpsychology is the study of mental phenomena in relation to cyber systems and cyber contexts. It is therefore a field of research that

[1] https://www.nato.int/nato_static_fl2014/assets/pdf/2020/5/pdf/2005-deepportal4-information-warfare.pdf

could produce knowledge that could be exploited as weapons for cognitive warfare, or, on the contrary, conceive ways of protecting against it.

F. Military Brain Science

According to Jin, Hou, and Wang [12], "Military Brain Science (MBS) is a cutting-edge innovative science […] based on the theories and technologies of medicine […], biology, physics, computer science, military science, and multiple other disciplines". It aims to monitor, protect, fight, repair, improve the brain. These are applications and tools that can be used for cognitive warfare, but remain focused on the military domain.

IV. Targets and Actors

One of the characteristics of Cognitive Warfare is that it can be conducted by anyone, against anyone, and at a distance.

A. Targets

We propose that three kinds of targets can be considered:

1) Large groups of people sharing a common characteristic (populations, opinion, ethnic or religious groups or minorities, etc.);

2) Small groups of people sharing a common goal (companies, teams, armed forces, etc.);

3) Critical individuals (decision-makers, politicians, military leaders or leaders of the various groups mentioned above, experts…);

People who have influence (critical individuals), either on a decision or on a group of people, can be particularly targeted, either directly or indirectly by attacking the large or small group they belong to, which will have repercussions on them (for example, a shift in public opinion can lead to a change of policy or a resignation). Bernal et al. [3] remind us as well that we should not forget "connectors, mavens, and salespeople", who "can be instrumental in the application of cognitive warfare".

As du Cluzel emphasizes, "any user of modern information technologies is a potential target. It targets the whole of a nation's human capital" [9].

B. Actors

Like targets, actors conducting Cognitive Warfare can be varied. Du Cluzel [9] points out that cognitive warfare and the advances in human sciences could potentially confer significant power to anyone who takes the trouble to study them, so that even isolated individuals or small groups can represent a major threat to democracies or military operations.

V. Cognitive Warfare and AI

Artificial Intelligence (AI) is bringing new tools facilitating cognitive warfare, that can amplify it and make it even more accessible and low cost, especially when it comes to fake news and disinformation diffusion.

Du Cluzel reminds us that fake news campaigns combine real and distorted information (misinformation), exaggerated facts and fabricated news (disinformation) [9].

Among these facilitating tools, Mad Scientist Laboratory [16] cites deepfakes, videos generated by artificial intelligence that can show a person reciting a speech he or she never actually gave: their danger is obvious, given that any influential personality can be made to say anything. They can be rendered even more realistic by technologies that imitate the tone of a person's voice and their accent [6]. The risk associated with AI-generated bodies and faces is less obvious, but just as real: it enables the creation of numerous fake accounts on social networks with people who do not exist, and makes it possible to humanize bots to give them more credibility. Generating the face of a person who does not exist is instantaneous, as can be seen at https://thispersondoesnotexist.com/. The last tool cited by Mad Scientist Laboratory is AI text generation, and this tool is brought up to date by the deployment of Chat-GPT in November 2022. This type of tool helps spread false information since it can write articles, posts and comments on social networks much faster and on a larger scale than a team of humans could. Thus, a single group could generate thousands or even millions of comments and posts on social networks, oriented to support or undermine a cause; and these actions would have "the potential to erode the relationship between governments and their citizens, provoking severe reactions throughout the world and leading people to question the very reality they believe" [16]. For example, the Twitter social media is host to many bots, which can "pursue malicious goals such as election interference and extreme propaganda" [10].

An example of the possible application of destabilization campaigns via fake news on social networks is the influence of elections. Russia is particularly active in this field, and "the Kremlin considers disinformation and information operations to be the most effective means of affecting political outcomes in other countries", seizing on "existing domestic political, social, or ethnic divisions and instrumentalizes them to change how voters think – and through that how they vote" [1].

VI. A Few Approaches to Protect Ourselves

Some examples of individual solutions exist against specific cognitive warfare actions. For example, some countermeasures against public influence and fake news on social media were listed: public education, communication about fake news, automatic and human moderation, debunks, legal regulations [1, 13, 27], etc.

Another way for protection and prevention is to use cognitive warfare defensively. Cognitive warfare tools can be used to educate the populations through media and social media [2], enhance cognitive readiness [19] or even augment soldiers' cognition [12]. We could also imagine decision-making tools taking into account cognitive biases and potential cognitive warfare aggressions.

The first step to organize an overall solution is to analyze the opponent and understand how they lead cognitive warfare strategies [21]. This would enable those under attack to be able to detect cognitive warfare offensives early and lift the fog of war [25]. Further research on this topic is necessary in order to build systematic solutions.

VII. Limitations and Criticisms of Cognitive Warfare

Certain destabilizing or influencing actions can be detected and countered: for example, a company implementing a cognitive warfare strategy towards its competitors or customers could be denounced by a data leak or a whistleblower, and this strategy would then backfire by damaging its public image.

Cognitive warfare may also face ethical challenges: it involves influencing the thinking and decision-making of a person or group of people without their knowledge. In this respect, it can be compared to nudges, which encourage the user of a system or tool to behave optimally [15]: it is also a tool of influence, but it is its use that determines its ethical characteristics. Indeed, if the nudge consists in pushing a consumer to buy more or at a higher price, it will be judged more negatively; but if it pushes him or her to behave more respectfully (according to the applicable societal and cultural norms), as with the example of the fly-shaped sticker in the toilet to encourage "better aim", it will be judged more positively. The same applies to cognitive warfare, which must be used in a reasonable and justifiable way.

ACKNOWLEDGMENT

This research is financially supported by the French Ministry of Defence - Defense Innovation Agency (AID).

REFERENCES

[1] O. Backes, and A. Swab, "Cognitive Warfare—The Russian Threat to Election Integrity in the Baltic States". Doctoral dissertation, Harvard University (2019).

[2] B. Battrawi, R. Muhtaseb "The Use of Social Networks as a Tool to Increase Interest in Science and Science Literacy : A Case Study of « Creative Minds »" Facebook Page. (2013).

[3] A. Bernal, C. Carter, I. Singh, K. Cao, and O. Madreperla, "Cognitive Warfare: An Attack on Truth and Thought". NATO and Johns Hopkins University: Baltimore MD, USA (2020).

[4] G.-D. Bobric, "The Overton Window : A Tool for Information Warfare". In J. Lopez, K. Perumalla and A. Siraj, ICCWS 2021 16th International Conference on Cyber Warfare and Security, pp. 20-27 (2021).

[5] F. Bulinge, "Renseignement militaire : une approche épistémologique". Revue internationale d'intelligence économique, 2, pp. 209-232 (2010). https://www.cairn.info/revue--2010-2-page-209.htm

[6] M. Chenouard, "L'intelligence artificielle peut-elle piéger votre mère en imitant votre voix ?" Courrier international. (2023). https://www.courrierinternational.com/video/video-l-intelligence-artificielle-peut-elle-pieger-votre-mere-en-imitant-votre-voix

[7] B. Claverie, B. Prébot, and F. du Cluzel, "Cognitive Warfare : La guerre cognitique". In B. Claverie, B. Prébot and F. du Cluzel (dir.), La Guerre Cognitique, p. 1, CSO (2021).

[8] B. Claverie, and B. Kowalczuk, "Chapitre 9 – Cyberpsychologie". In B. Claverie, B. Prébot & F. du Cluzel (dir.), La Guerre Cognitique, pp. 9.1-9.5, CSO (2021).

[9] F. du Cluzel, "Cognitive Warfare". Innovation Hub (2020).

[10] S. Feng, H. Wan, N. Wang, J. Li, and M. Luo, "TwiBot-20: A Comprehensive Twitter Bot Detection Benchmark". In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 4485-94 (2021). https://doi.org/10.1145/3459637.3482019.

[11] C. Harbulot, "De la légitimité de la guerre cognitive. Revue internationale et stratégique", 56(4), pp. 63 67 (2004).

[12] H. Jin, L.-J. Hou, and Z.-G. Wang, "Military Brain Science – How to influence future wars". Chinese Journal of Traumatology, 21(5), pp. 277-280 (2018). https://doi.org/10.1016/j.cjtee.2018.01.006

[13] Johns Hopkins University & Imperial College London. "Sensibilisation et résilience, les meilleures armes contre la guerre cognitive ". NATO Review, 2021. https://www.nato.int/docu/review/fr/articles/2021/05/20/sensibilisation-et-resilience-les-meilleures-armes-contre-la-guerre-cognitive/index.html#:~:text=Dans%20la%20guerre%20cognitive%2C%20l,mais%20%C3%A9galement%20sur%20leurs%20actes.

[14] A. Jousset, and K. Bolchakova, "Wagner, l'armée de l'ombre de Poutine". Capa Presse (2022).

[15] G. Loewenstein & N. Chater. "Putting Nudges in Perspective". *Behavioural Public Policy* 1, nº 1 (2017): 26-53. https://doi.org/10.1017/bpp.2016.7.

[16] Mad Scientist Laboratory contributors. "149. The Death of Authenticity : New Era Information Warfare." (2019, May 30). https://madsciblog.tradoc.army.mil/149-the-death-of-authenticity-new-era-information-warfare/

[17] I. Malin, "Cyberattaques : comment l'Ukraine a failli perdre la guerre avant même l'invasion russe." (2022, 9 October).

[18] P. Montocchio "Avant-propos par le directeur adjoint du Collaboration Support Office (CSO) STO". In B. Claverie, B. Prébot and F. du Cluzel (dir.), La Guerre Cognitique, pp. vii-viii, CSO (2021).

[19] J. E. Morrison & J. D. Fletcher, "Cognitive Readiness". Institute for Defense Analyses. (2002)

[20] B. Norton, "Behind NATO's 'cognitive warfare': « Battle for your brain » waged by Western militaries". The Grayzone (2021). https://thegrayzone.com/2021/10/08/nato-cognitive-warfare-brain/

[21] K. Orinx, and T. Struye de Swielande, "La guerre cognitive – Pourquoi l'Occident pourrait perdre face à la Chine ?" In B. Claverie, B. Prébot and F. du Cluzel (dir.), La Guerre Cognitique, pp. 8.1-8.7, CSO (2021).

[22] O. Pinard Legry, "Neurosciences et sciences cognitives : Comment se préparer à la guerre des cerveaux ?" Revue Défense Nationale, N° Hors-série(HS3), pp. 58-76 (2022).

[23] J. Prier, "Commanding the trend : Social media as information warfare". In *Information Warfare in the Age of Cyber Conflict*, pp. 88-113, Routledge (2020).

[24] "Réforme des retraites : Des députées RN cibles de messages d'intimidation". Le Monde.fr. (2023, février 7). https://www.lemonde.fr/politique/article/2023/02/07/reforme-des-retraites-des-deputees-rn-cibles-de-messages-d-intimidation_6160845_823448.html

[25] Weldon, A. (2021). *Bytes not bombs : Student team works with NATO to define, track cognitive warfare attacks*. Johns Hopkins University - The Hub. https://hub.jhu.edu/2021/10/06/cognitive-warfare-attacks/

[26] Wikipedia contributors. "Syndrome de La Havane". In Wikipédia (2022). https://fr.wikipedia.org/w/index.php?title=Syndrome_de_La_Havane&oldid=197673790

[27] M. Wunder, "Chapitre 7 – Les narrations submergent le monde". In B. Claverie, B. Prébot and F. Du Cluzel (dir.), *La Guerre Cognitique* (p. 7.1-7.4). CSO (2021).

Red Team LLM: towards an adaptive and robust automation solution

Christophe Genevey-metat
R&D AI / Cyber Team
Silicom
Rennes, France
cgeneveymetat@silicom.fr

Dorian Bachelot
R&D AI / Cyber Team
Silicom
Rennes, France
dbachelot@silicom.fr

Tudy Gourmelen
R&D AI / Cyber Team
Silicom
Rennes, France
tgourmelen@silicom.fr

Adrien Quemat
R&D AI / Cyber Team
Silicom
Rennes, France
aquemat@silicom.fr

Pierre-Marie Satre
R&D AI / Cyber Team
Silicom
Rennes, France
pmsatre@silicom.fr

Loïc Scotto Di Perrotolo
R&D AI / Cyber Team
Silicom
Rennes, France
lscottodiperrotolo@silicom.fr

Maximilien Chaux
R&D AI / Cyber Team
Silicom
Rennes, France
mchaux@silicom.fr

Pierre Delesques
R&D AI / Cyber Team
Pr0ph3cy
Guyancourt, France
pdelesques@pr0ph3cy.com

Olivier Gesny
R&D AI / Cyber Team
Pr0ph3cy
Rennes, France
ogesny@silicom.fr

***Abstract*—Artificial intelligence has become really popular in recent years, especially its embedding in cybersecurity applications. Today, studies have shown that reinforcement learning agents are able to find the optimal sequence of actions in order to attack a network. However, these agents are often over-trained and can neither adapt nor be robust to different networks from the ones they were trained on. We propose a new agent based on a zero-shot approach that adapts itself to any given network and that is robust to parameters and objectives changes without requiring another training phase. We introduce a new metric that better measures the ability of agents to attack a network without prior knowledge. In this paper, we also discuss about the first steps towards explainability for our model and its future improvements.**

***Index Terms*—pentesting automation, attack simulation, zero-shot classification, transformers, large language model, adaptability, robustness, explainability**

I. Introduction

Pentesting is an operation that consists in discovering the weaknesses of a network that an attacker could use to harm a company. In the research field, pentesting is often seen as an optimisation problem where the goal is to compromise a machine inside a network within a minimum number of actions. The actions are unit attacks that can be chosen by the agent. Several researchers [1], [3], [11], [14], [16] have already studied the use of reinforcement algorithms to solve this optimisation problem and several simulators, such as NASim [6], or Cyborg [8], have been developed to facilitate working on such algorithms. In a real-world situation, when an intruder attacks a network, having access to the entire network's arrangement beforehand is quite rare. Indeed, each network has its own topology with its own layout of subnetworks (subnets) and machines. As a result, an attacker needs to navigate between exploring the network in search of information about the target, exploiting machines and in some cases, bypassing firewall's restrictions. Thus, we believe that pentesting cannot be considered like a simple optimisation problem due to the presence of changing sets of actions and heavy interdependence between obtained answers and new actions at each step. The simulator NASim [6] developed by J. Schwartz is one of the most complex simulators that exist and represents well certain aspects and constraints that a real attacker may encounter. We developed a variant of NASim that can produce a natural textual representation of the network used by our model. We also use this modified NASim to calculate our own metric that better represents the agent's ability to use discoStovered information before exploiting an unknown machine.

In this paper, we use a pre-trained zero-shot classification model [15]. We have developed a specific textual observable environment to provide the model with a better level of adaptability compared to other reinforcement learning algorithms. Thus, this model is adaptive and able to attack other networks, unseen during the training phase. This model uses textual observables to take the most promising action. In these textual observations, we provide the goal of the challenge, some rules that the model has to follow to complete the challenge and information about the state of the targeted information system (e.g. machine state). We have also created a naive metric for the training and evaluation phase in order to know if the agent has taken the correct action thanks to its past knowledge, meaning that it was based upon useful information gathered prior to the execution. Or if it did by chance, implying that the attack is successful but without the agent exploiting the gathered information. For example, this naive metric allows the agent to be rewarded if it discovers that a certain vulnerable service is running on a machine and attacks this machine. We compare the performances of our reinforcement learning (RL) zero-shot model with other reinforcement learning algorithms (DQN, PPO recurrent, and CLAP) and discuss the results with our own metric further in the paper. In addition to this comparison, we also provide a preliminary explainability of our model based on the Shap

value [5]. We demonstrate our agent's confidence based on the observable textual environment (feasible actions included), as well as the importance of certain words in the context that allow the agent to take the right action. Since our model uses textual information, it is possible to extract a certain degree of explainability. Yet still limited, it opens up the discussion for more.

II. Related works

Schwartz and Kurniawati [6] proposed a benchmark environment to train and evaluate RL agents in the context of pentesting. Their benchmark environment called "NASim" allows them to simulate a network with various subnets and several machines per subnet. They include services, processes and operating systems in the machine information. Their environment also includes some restrictions of services to support several types of firewall configurations. They propose a list of actions that an RL agent can perform. Each action has a certain cost (due to its level of furtiveness) and a certain probability of success (for a non-deterministic aspect). This environment is a pure simulation but it includes some constraints found in the real world, these are discussed in section III-A.

Another benchmark environment exists called CybORG [9] developed for the CAGE (Cyber Autonomy Gym for Experimentation) challenges from the Department of Defense of the Australian government. This environment is better equipped than NASim and able to support simulations as well as emulation. It is even cited in the gap considerations page of NASim as an alternative for more thorough tests. However, we chose to go with NASim for several practical reasons. Firstly, the emulation part of CybORG was of no interest for us at this point and is not open sourced. Secondly, it was easier to compare our model to existing techniques. Indeed, most of the RL algorithms published up to this day use NASim and its predefined topologies and scenarios for benchmarking.

In addition to the proposed NASim environment, Schwartz [6] shows the performances of classical RL agents (DQN, Tabular Q-learning) on miscellaneous simple scenarios. They establish the performance of these RL agents against the number of machines in the network. They demonstrate how the performance decreases as network complexity increases, and that after a certain number of machines, a classical RL agent obtains a negative mean episodic reward.

Zhou and Liu [16] propose an improvement of the deep Q-network named NDSPI-DQN to solve the issue of sparse rewards and large action space problems. Their model uses soft Q-learning, duelling architecture, prioritised experience replay, and intrinsic curiosity to improve the exploration efficiency. They show that their model converges better than the baseline DQN, and can achieve a positive average episodic return, while the original DQN achieves a negative average episodic return.

Yizhou and Xin [14] propose a variant of the PPO algorithm called CLAP that can handle multi-objective reinforcement learning in a pentest context. They study the multi-objective in order to provide a diversity of attacks and paths that can compromise the network's security. CLAP uses a coverage mask mechanism that allows the model to keep track of previous actions taken in the past. This mechanism encourages the model to trigger an action based on both the previous action and the current observation. The Clap model is trained on three NASim scenarios described in table I. These three scenarios have different topologies and objectives. They show that their model converges quickly to the optimal sequence on the three scenarios during the training phase compared to other algorithms (DQN, Improved-DQN, HA-DQN). However, the authors only show the performance during the training phase and not for an evaluation phase. Thus, it raises questions about the adaptability of their model regarding other networks and during an evaluation phase.

TABLE I
NETWORK SCENARIOS USED BY CLAP MODEL.

	Subnets	Hosts	OS	Services	Processes
Small-linear	7	8	2	3	2
Medium	6	16	2	5	3
Large	8	23	3	7	3

In section III, we present the challenge for a RL agent to attack a network without prior knowledge, the process of decision-making, the observable environment developed for our model, and our own metric. In section IV, we present the performance of our model compared to other RL agents (DQN, PPO recurrent, and CLAP), and a preliminary study to observe the impact of certain words in the decision-making process.

III. Preliminary

In this section, we present the challenge of pentesting, the process of decision-making of our agent inside an adapted NASim environment, and our new metric for the training and evaluation.

A. Challenge of pentesting

From our point of view, the role of a Red Team is complex and is not limited to compromising a specific machine in a known information system as quickly as possible.Indeed, a RL agent does not have any prior knowledge about the target network. It operates on a partially observable environment, and its action space expands because a list of new actions appears when it discovers a machine.

We identify other constraints that an RL agent must take into account due to no prior knowledge about the target network. These following constraints are not specific to a RL agent and also exist for other machine learning agents that perform pentesting:

- Machine choice: the agent must be able to select the right machine to attack. If it discovers the target machine then it has to focus onto it, otherwise it should focus on the other machines of the network. For example, the machine

identifier may change from one network to another, as may its operating system or services.
- Attack choice: the agent needs to be able to select the right attack. If it doesn't have enough information to do so, it should keep going with the discovery stage; but if it does have enough, it should exploit the machine.
- Probability of success: an action has a certain probability of success. If the agent selects an action that fails, it needs to try again or select another action. The probability of success can vary from one network to another.
- Restriction of services: an action can fail due to some restrictions from the system. The agent needs to be able to select another path in order to compromise the machine, should a restriction happen. In addition, the agent must be able to make errors in order to discover this restriction. The restriction of services can also vary from one network to another.
- Choice of parameters: the agent must have the ability to select the right action based on the information discovered on the machine and the privilege levels available after the attack. The privilege levels on one machine can vary from one network to another.

These constraints may seem simple to manage for a human operator, but represent a challenge for RL agents because most of the time, RL agents will just repeat the optimal sequence they learned during training. If the network changes and these constraints also change, the RL agent will not be able to act correctly. This is why we believe that pentesting cannot be considered as a simple optimisation problem. We decided to choose the NASim environment because it includes all of these constraints. It can therefore be seen as a first pentesting environment for RL agents to work on.

B. Decision-making

Our approach is based upon the work of *Yin et al.* [15]. They propose a method for using a pre-trained sequence-to-sequence model and adapt it to a zero-shot classifier. Their model takes as input a sequence to be classified and generates a hypothesis from each candidate label. Their method can be used with different large pre-trained models such as BART [4] or Roberta [2] which are Transformers networks. We chose to focus on the BART model and slightly adapt their approach to play with a reinforcement learning environment. The Figure 1 illustrates the path and reasoning of our model when playing in reinforcement learning environment, and especially in the NASim environment [6]. Our model, in order to make a decision, works with two main types of sub-decision: it first chooses the target machine to attack, then it selects the action to perform on it. The "Overall context" is a description of the goal to achieve supported by some advice. The "machine available" is a description of the status of each machine. It represents the candidate labels. Thus, our model selects one of these candidates based on the overall context. After selecting the machine, our model must choose the action to perform on it. The "Machine's context" is a description of the current goal and the information found on the machine (during the previous iteration). The "Available actions" is a description of the available attacks that can be launched by our model. It is the new candidate labels that our model will select based on the context of the machine. After selecting the action, it is performed on the NASim environment and new "Overall context", "machine available", "Machine's context" and "Available actions" are generated.

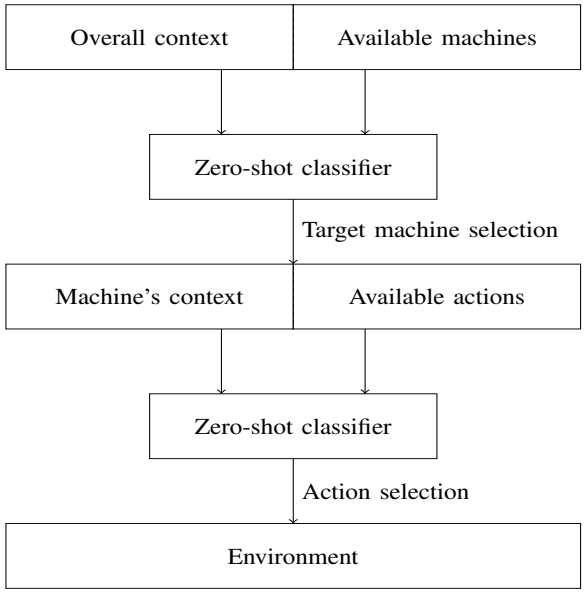


Fig. 1. Decision path taken by our model

C. Textual observables for decision support

In order to help our zero-shot model to select the right target machine and the correct action, we create a certain global context, a machine context, an action description and some textual observables for decision support.

1) Overall context: In the overall context, we store information about the potential path to the goal, a description of the goal to reach, and rules that the zero-shot must follow. The potential path to the goal is regularly updated by removing discovered and exploited machines from the list of potential paths. This textual observable helps the agent to focus solely on machines not yet exploited. The objective is also updated if it is composed of several sub-goals and if one of them is achieved. Therefore, the agent is able to focus on the second objective when the first one is accomplished. Unlike the other textual observables, the rules that the zero-shot must follow do not change. This element helps the agent to find another path when a restriction appears, it also gives greater confidence in the agent's decision-making. In section IV-B, we will talk more about this confidence in the decision making process.

Example of 'Overall context': *B,C,D,E,F are potentially linked to F. The goal is to get privileges on F while remaining undetected. We encourage to keep attacking machines where you do not have full access.*

2) Available machines: In the "Available machines" part, there is a descriptive list of all the discovered machines' statuses. The status of the machine is updated on certain conditions. Table II describes the conditions that make the candidate labels changes. The conditions and labels we have developed give the agent a better representation of the environment helping it in making the right decision at all times. We have chosen these labels in order to obtain decisions as deterministic as possible. The impact of the words used in these labels will be developed in section IV-B.

Example of 'Available machines': *A has already been attacked, B is a new machine, C is a new machine, D is a new machine, E is a new machine, F is one of the target machine.*

Conditions	Candidate labels (Actions)
Agent does not execute any action on A	A is a new machine
Agent executes at least one action on A	A has already been attacked
Agent discovers that A is the target machine	A is one of the target machine
Agent obtains root privileges on A	A is totally corrupted
Agent executes at least one action and receives a permission error	A is a dead end

TABLE II
CONDITIONS TO UPDATE THE CANDIDATE LABELS THAT CAN BE PRESENT IN THE LIST OF 'AVAILABLE MACHINE'

3) Machine context: Inside the machine context we have information about the goal, the services, the processes, and the operating system present on the machine. It also states if the machine is compromised or not. The goal information is updated as in the overall context. Most of this information is also used by the other RL agents (DQN, CLAP) from the state-of-the-art papers, except for the goal information part. Table III describes the observables that represent all information about a machine, and how it is updated after the agent triggers an action. We only illustrate the update via a specific sequence to give an overview, and to highlight the impacted data.

Example of 'Machine context': *The goal is to get privileges on F while remaining undetected. No service found on F. No operating system found on F. No process found on F. No privilege obtained on F.*

4) Available actions: In the available actions, we have information about actions that can be executed on the targeted machine. Table IV describes each action present in our scenarios of NASim: Tiny, Small-linear, and Medium (defined in the Table V). Currently, we have created 12 textual labels that allow playing on different scenarios of NASim but not all. Some of these actions also target an operating system (Linux, or Windows) in addition to the machine, and we take this into account in our labels. We have also added information to some actions for the agent to execute them when a certain status of a machine appears. This is particularly true for the following actions: "process_scan", "subnet_scan", "pe_daclsvc", "pe_tomcat", and "pe_schtask" which will be executed once the machine is infected.

Conditions	Observables
The agent does not execute any action on A	No service found on A. No operating system found on A. No process found on A. No privilege obtained on A.
The agent executes 'service_scan' on A	**The service ssh is running on A**. No operating system found on A. No process found on A. No privilege obtained on A.
The agent executes 'os_scan' on A (in addition to the previous)	The service ssh is running on A. **The operating system linux is running on A**. No process found on A. No privilege obtained on A.
The agent executes 'e_ssh' on A (in addition to the previous)	The service ssh is running on A. The operating system linux is running on A. No process found on A. **A is finally infected**.
The agent executes 'process_scan' on A (in addition to the previous)	The service ssh is running on A. The operating system linux is running on A. **The process tomcat is running on A**. A is finally infected.

TABLE III
CONDITIONS TO UPDATE THE OBSERVABLES THAT CAN BE PRESENT IN A 'MACHINE CONTEXT'

NASim Actions	Candidate labels (Actions)
service_scan	reveal service on A
os_scan	reveal operating system on A
process_scan	reveal process by infecting A
subnet_scan	discover subnet by infecting A
e_ftp [linux,windows]	exploit ftp [linux,windows] on A
e_http [linux,windows]	exploit http [linux,windows] on A
e_ssh [linux,windows]	exploit ssh [linux,windows] on A
e_samba [linux,windows]	exploit samba [linux,windows] on A
e_smtp [linux,windows]	exploit smtp [linux,windows] on A
pe_daclsvc	get highest privileges by infecting A with daclsvc windows
pe_tomcat	get highest privileges by infecting A with tomcat linux
pe_schtask	get highest privileges by infecting A with schtask windows

TABLE IV
LABELS USED TO REPRESENT THE NASIM ACTIONS

Example of 'Available actions': *reveal service on F, discover subnet by infecting F, reveal process by infecting F, exploit ssh linux on F, exploit http on F, exploit ftp windows on F, get highest privileges by infecting F with tomcat linux, get highest privileges by infecting F with daclsvc windows.*

D. Evaluation metric

As presented in section III-A, we suggest that measuring the performance of a RL agent based on the number of actions done to compromise the target machine is not an efficient metric because it does not encourage actions that reflect the behaviour of a human pentester confronted with an IS that he does not know. Indeed, pentesters tend to want to be as discreet as possible and to obtain as much useful information about a machine as possible before attacking it. We choose to evaluate the ability to attack a network with a new metric that encourages the RL agent to discover the necessary information before attacking a machine.

At the end of the challenge, we return the basic reward plus a bonus. Equation 1 describes this new metric which allows to better evaluate the performance of our model during the training and evaluation phase. The basic reward called R_{base} is the sum of the cost of all the actions carried out. The bonus is the sum of A_c^t (multiplied by a factor α) which is the one-hot incorporation of the action taken by the agent at time t after having discovered the right information. α is a configurable factor that compensates for the cost of discovery actions. RL agent using information from discovery actions to exploit machines obtain reward bonuses. This factor must be higher than 3 because the attacker needs to discover at least three elements per machine: operating system, services, processes and each discovery cost -1. This metric is used with all agents (CLAP, DQN, PPO, Zero-shot) and we have a safeguard that prevents discovery actions that have already been positively rewarded from being rewarded again.

$$R_{final} = R_{base} + \sum_{i=1}^{t} A_c^t * \alpha \tag{1}$$

IV. Contributions

A. Experiments on NASim

In this section, we present the performances of our zero-shot model compared to the PPO recurrent network and the CLAP model.

TABLE V
NETWORK SCENARIOS USED BY OUR MODEL.

	Subnets	Hosts	OS	Services	Processes
Tiny	4	3	1	1	1
Small-linear	7	8	2	3	2
Medium	6	16	2	5	3

We also present the performances of our model with our new metric used during the evaluation phase. Refer to Appendix 10 for a comparison of the performance of our model against other Reinforcement Learning agents, using the traditional reward.

For the experiments, we choose to compare the DQN, PPO recurrent, and CLAP with our model. The DQN and CLAP were chosen because they are mentioned in several state-of-the-art papers [6], [14]. We also decided to add the PPO recurrent since we wanted a classical RL agent that can handle partially observable environment. We had to train the other models in order to compare them with ours. This training took 100,000 steps for the DQN and PPO recurrent models, and 5,000,000 for the CLAP model. It is important to note that our model did not require any training at all because it is a pre-trained model, therefore we haven't re-trained it on the NASim environment. We have noticed that the DQN and PPO recurrent do not converge after 100,000, this is why the training steps are set to this value. For CLAP, we took the same hyperparameters as the authors chose [14].

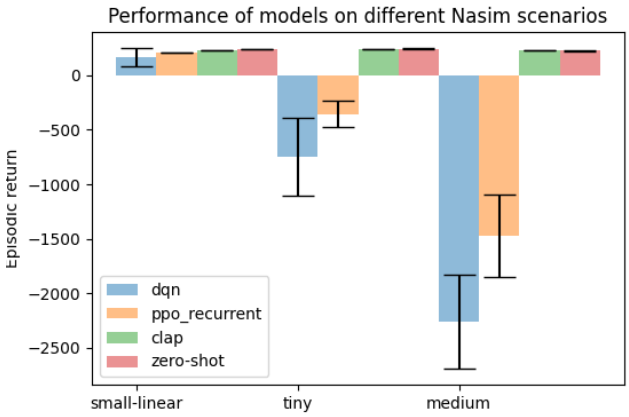


Fig. 2. Comparison of performances from RL agents (DQN, CLAP, PPO recurrent and zero-shot) trained and evaluated with the new evaluation metric

Figure 2 shows the performance of our zero-shot model compared to other RL agents (CLAP, PPO recurrent). The episodic return is the final reward given to the agent at the end of the challenge. This reward is computed with the new metric evaluation introduced in the previous section. We compare all algorithms on three scenarios (tiny, small-linear, and medium) using a mean episodic return. It is the mean value obtained after five runs.

In the following results 2, we can see that the DQN, PPO recurrent and CLAP obtain good results. However, we would like to remind that these models are not adaptive and they are evaluated on the same network they trained on. Whereas our zero-shot approach does not require training and therefore does not know the targeted network in advance.

For the tiny scenario (on the left-hand side of the figure 2), we observe that the four models have positive episodic returns. The DQN model has a mean episodic return of 163, the PPO model gets a mean episodic return equal to 207, the CLAP model obtains a mean episodic return of 230, and the zero-shot model reaches a mean episodic return equal to 239. Thus, our model outperforms the DQN, PPO recurrent and CLAP network, even if these models have seen the network during the training phase.

For the small-linear scenario (on the middle of the figure 2), we observe that only the CLAP and zero-shot models have positive episodic returns. The DQN model reaches a mean

episodic return of –746, the PPO model has a mean episodic return equal to -355, the CLAP model obtains a mean episodic return of 235, and the zero-shot model gets a mean episodic return equal to 244. The DQN has difficulties to converge due to the partially observable environment and the complexity of the network. The PPO recurrent obtains a better score than the DQN thanks to its recurrent layer that better handles partially observable environment. Finally, we observe that our model is 61 points better compared to the CLAP model.

For the medium scenario (on the right-hand side of the figure 2), we also observe that only CLAP and zero-shot models have positive episodic returns. The DQN model reaches a mean episodic return of -2260, the PPO model obtains a mean episodic return equal to -1471, the CLAP model has a mean episodic return of 234, and the zero-shot model gets a mean episodic return equal to 225. We observe that CLAP outperforms our model, our model is -9 points worst compared to the CLAP model. In this case, CLAP performs better, which is not surprising, because it should be pointed out that it saw the network during its training phase, whereas our model did not.

These results show that our zero-shot approach outperforms other RL agents (PPO and CLAP) in terms of mean episodic returns. We can conclude that our model can handle different networks without needing a new training.

B. Towards explainability

We are aware that the work we present cannot lead to a complete explainability of our model. However, this work has allowed us to better understand the importance of the words that are given to the algorithm. The choice of words also helped in the decision making process to get more accurate and deterministic answers. This is why we present, in this section, a preliminary study on the importance of words in decision making.

We start by looking at the confidence that our model gives to each candidate labels. We then look at the Shap value whose interest is to explain the influence of inputs on model outputs. In our case, we can thus see the influence of the input words on the decision-making of the actions (selection of the machine and selection of the action).

1) Machine selection: In this section, we will show how the words in the list of available machines impact the decision of our models. In order to illustrate this, we have created five distinct contexts called ”Context 1” to ”Context 5” and three candidate labels per context. These new contexts and candidate labels are illustrated and described in the Table VI and IX. In the Figure 3, we can observe the probability of each action A1, B1, C1 for the different contexts. In the context 1, we can select three new machines represented by the three actions A1, B1, C1. There is no advantage in selecting one over the other, this is why we can observe close probabilities between actions A1, B1, and C1. Thus, our model decides to choose the machine A (by selecting the action A1). In the context 2, the agent has previously launched one action on the machine A. Therefore, we can observe that the action A1

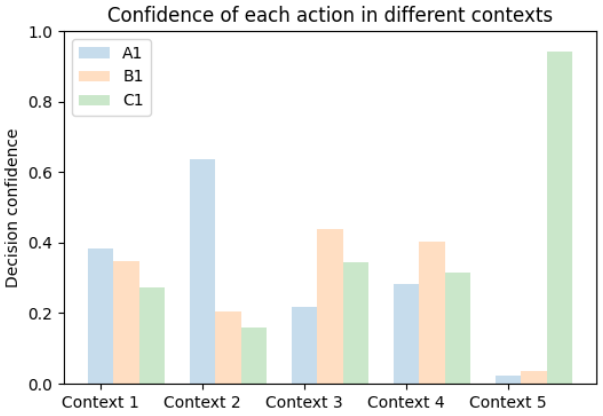


Fig. 3. Decision confidence for different contexts about selecting the machine

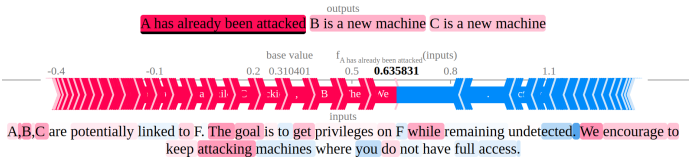


Fig. 4. Shap value of the action ”A has already been attacked”

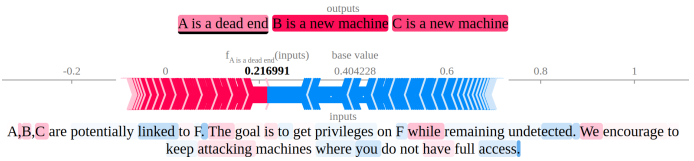


Fig. 5. Shap value of the action ”A is a dead end”

gains in confidence in the decision (compared to the previous context). With the rule given in the context (”We encourage to keep attacking machines where you do not have full access.”), the confidence in the selection of machine A increases. In the context 3, the word ”dead end” is used when the agent discovers a restriction of service. Thus, we can observe that the confidence about A1 is the lowest. We found that the use of particular words (”dead end”, ”arrested”, or ”stop”) in the observation of an action has a negative effect and reduces the probability of choosing that action. Figures 4 and 5 show the Shap value and confirm this effect. We observe that without the term ”dead end”, many keywords from the context are used (red colour) however when the sequence ”is a dead end” is given, no more words from the context is used. Thus, when our agent discovers a restriction, we include this word into the context in order to avoid that our agent gets blocked and therefore selects another machine. This behaviour is confirmed in the context 3, where our agent prefers to select the machine B since the machine A is a dead-end. In the context 4, the action B1 is selected because the machine A is totally corrupted. Thus, our agent selects a new machine such

as B or C. Finally, in the context 5, our agent discovered that the target machine is C, the confidence is then higher for the action C1.

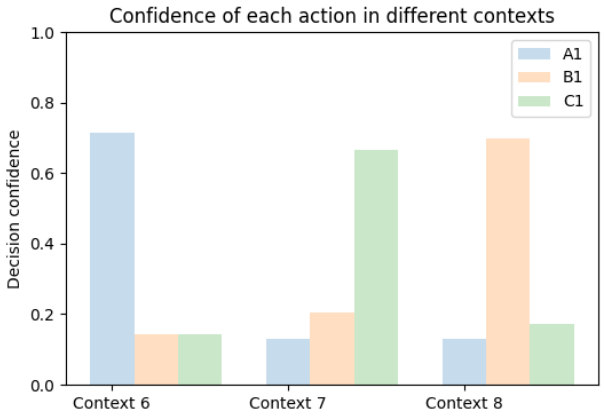


Fig. 6. Confidence in decision for different contexts when selecting actions

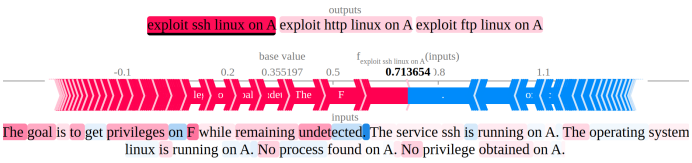


Fig. 7. Shap value of the action ”exploit ssh linux on A”

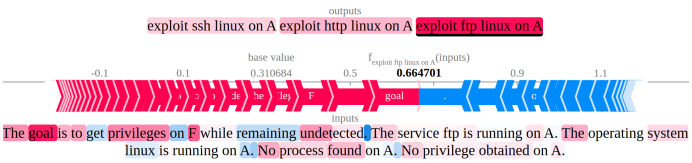


Fig. 8. Shap value of the action ”exploit ftp linux on A”

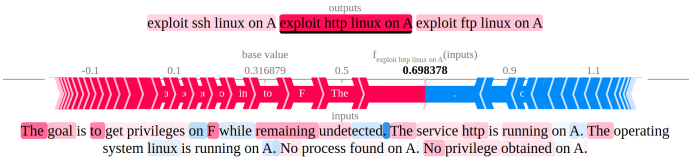


Fig. 9. Shap value of the action ”exploit http linux on A”

2) Services selection: In this section, we show how the words that correspond to the name of service impact the decision of our model. In order to illustrate this, we have created three other contexts called ”Context 6”, ”Context 7”, and ”Context 8” and three candidate labels per context. These new contexts and candidate labels are illustrated and described in Table VIII and IX. In the Figure 6, we can observe the probability of each action A1, B1, C1 for the different contexts. In the context 6, the service ssh is running on the machine. The action with the highest probability is A1, ”exploit ssh”, since the model knows that the associated service is running. In the other contexts, we can observe the same results. So if we change the name of the service running on the machine, the action exploiting this service will have a higher probability to be picked than the others. We also confirm that our model uses the name of services available in the context with the different Shap values present in the figures 7, 8, and 9. In the Figures 8 and 9, we observe that the word ”http” and ”ftp” are used to take a decision.

V. Discussion / Future works

In this paper, we have presented a more robust and adaptive approach based on a zero-shot classification that outperforms the other RL agents (DQN, PPO recurrent, and CLAP) in the reference NASim environment according to our new metric. We have made sure that our solution for building the environment is easily adaptive to any simulated network topology. We already have first conclusive results when exploiting the model in a realistic cyberrange environment. In order to improve the generalisation of the model, the next step is to test it in a more realistic and complex environment. We also want to improve our approach by using a more powerful AI architecture. Indeed, we plan on using Text2Text generation such as Llama [10], Huggingchat [7], FLAN [12], or Bloom [13] models instead of a zero-shot classifier with the BART model. We hope that using one of these models will reduce the amount of help we currently give in the observable environment. Doing so would also improve handling more complex pentesting challenges with more constraints, restrictions and services.

Our version of NASim and our naive evaluation metric will be available at the following URL: https://github.com/silicom-hub/zero-shot-pentesting-paper.

References

[1] Alessandro Confido, Evridiki V. Ntagiou, and Marcus Wallum. Reinforcing penetration testing using ai. In *2022 IEEE Aerospace Conference (AERO)*, pages 1–15, 2022.

[2] Naman Goyal, Jingfei Du, Myle Ott, Giri Anantharaman, and Alexis Conneau. Larger-scale transformers for multilingual masked language modeling. *CoRR*, abs/2105.00572, 2021.

[3] Zhenguo Hu, Razvan Beuran, and Yasuo Tan. Automated penetration testing using deep reinforcement learning. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 2–10, 2020.

[4] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019.

[5] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[6] Jonathon Schwartz and Hanna Kurniawati. Autonomous penetration testing using reinforcement learning. *CoRR*, abs/1905.05965, 2019.

[7] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface, 2023.

[8] Maxwell Standen, David Bowman, Son Hoang, Toby Richer, Martin Lucas, Richard Van Tassel, Phillip Vu, Mitchell Kiely, KC C., Natalie Konschnik, and Joshua Collyer. Cyber operations research gym. https://github.com/cage-challenge/CybORG, 2022.
[9] Maxwell Standen, Martin Lucas, David Bowman, Toby J. Richer, Junae Kim, and Damian Marriott. Cyborg: A gym for the development of autonomous cyber agents, 2021.
[10] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.
[11] Khuong Tran, Ashlesha Akella, Maxwell Standen, Junae Kim, David Bowman, Toby Richer, and Chin-Teng Lin. Deep hierarchical reinforcement agents for automated penetration testing, 2021.
[12] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners, 2022.
[13] BigScience Workshop, :, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, and Daniel Hesslow. Bloom: A 176b-parameter open-access multilingual language model, 2023.
[14] Yizhou Yang and Xin Liu. Behaviour-diverse automatic penetration testing: A curiosity-driven multi-objective deep reinforcement learning approach, 2022.
[15] Wenpeng Yin, Jamaal Hay, and Dan Roth. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. *CoRR*, abs/1909.00161, 2019.
[16] Shicheng Zhou, Jingju Liu, Dongdong Hou, Xiaofeng Zhong, and Yue Zhang. Autonomous penetration testing based on improved deep q-network. *Applied Sciences*, 11(19), 2021.

VI. Annexe

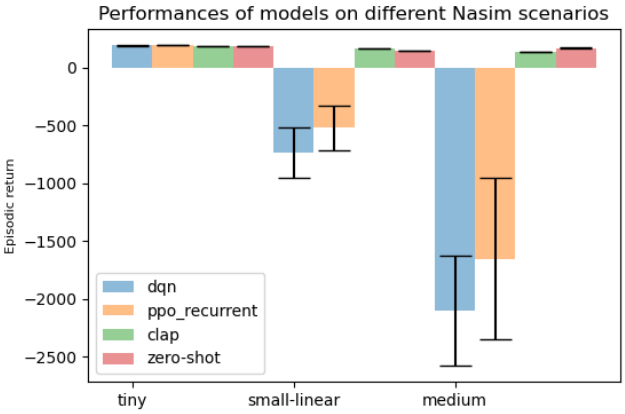


Fig. 10. Comparison of performances from RL agents (DQN, CLAP, PPO recurrent and zero-shot) trained and evaluated with the NaSim metric evaluation

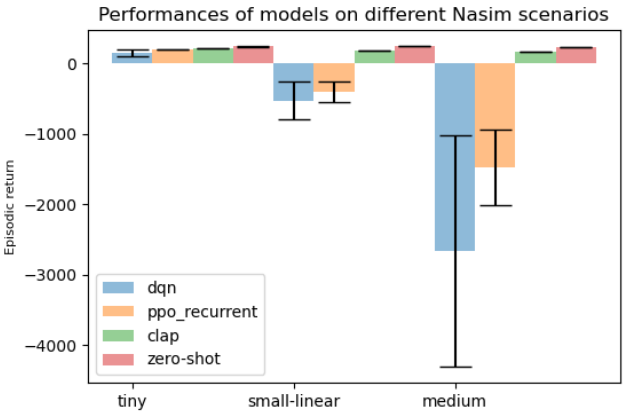


Fig. 11. Comparison of performances from RL agents (DQN, CLAP, PPO recurrent and zero-shot) trained with the NaSim metric and evaluate with the new metric evaluation

Context	Description
Context 1, 2, 3, 4, 5	A,B,C are potentially linked to F. The goal is to get privileges on F while remaining undetected.We encourage to keep attacking machines where you do not have full access.

TABLE VI
List of contexts for machine selection

Context	Candidate label	Description
Context 1	A1	A is a new machine
	B1	B is a new machine
	C1	C is a new machine
Context 2	A1	A has already been attacked
	B1	B is a new machine
	C1	C is a new machine
Context 3	A1	A is a dead end
	B1	B is a new machine
	C1	C is a new machine
Context 4	A1	A is totally corrupted
	B1	B is a new machine
	C1	C is a new machine
Context 5	A1	A is totally corrupted
	B1	B is a new machine
	C1	C is one of the target machine

TABLE VII
LIST OF CONTEXTS AND CANDIDATE LABELS FOR MACHINE SELECTION

Context	Description
Context 6	The goal is to get privileges on F while remaining undetected. The service ssh is running on A. The operating system linux is running on A. No process found on A. No privilege obtained on A.
Context 7	The goal is to get privileges on F while remaining undetected. The service ftp is running on A. The operating system linux is running on A. No process found on A. No privilege obtained on A.
Context 8	The goal is to get privileges on F while remaining undetected. The service http is running on A. The operating system linux is running on A. No process found on A. No privilege obtained on A.

TABLE VIII
LIST OF CONTEXT FOR MACHINE SELECTION

Context	Candidate label	Description
Context 6	A1	exploit ssh linux on A
	B1	exploit http linux on A
	C1	exploit ftp linux on 1
Context 7	A1	exploit ssh linux on A
	B1	exploit http linux on A
	C1	exploit ftp linux on A
Context 8	A1	exploit ssh linux on A
	B1	exploit http linux on A
	C1	exploit ftp linux on A

TABLE IX
LIST OF CONTEXT AND CANDIDATE LABEL FOR MACHINE SELECTION

Un système d'apprentissage ensembliste explicable pour détecter des attaques réseau inconnues

Céline Minh*†, Kevin Vermeulen†, Cédric Lefebvre*, Philippe Owezarski†, William Ritchie*
*Custocy, Toulouse, France. Email: {cminh, clefebvre, writchie}@custocy.com
†LAAS-CNRS, Université de Toulouse, CNRS, INSA, Toulouse, France. Email: {celine.minh, kevin.vermeulen, owe}@laas.fr

***Résumé*—L'apprentissage automatique est une technologie prometteuse pour les systèmes de détection d'intrusions. Il existe une grande variété d'algorithmes d'apprentissage automatique dont les résultats semblent complémentaires, mais déterminer quel résultat est correct dans un cas spécifique est difficile car les algorithmes ne sont pas tous explicables. Cet article introduit un système conçu pour être explicable qui reconstruit des schémas d'attaques à partir des sorties d'un ensemble de détecteurs d'anomalies non supervisés et les présente aux analystes de sécurité pour les aider à interpréter chacune des détections.**

***Mots-clés*—Sécurité réseau, détection d'anomalies non supervisée, IA explicable, apprentissage ensembliste, CNN**

I. Introduction

L'apprentissage automatique (ML) est une technologie prometteuse pour les systèmes de détection d'intrusion réseau (NIDSs) car il permet de détecter des attaques sur de grandes quantités de données. Un inconvénient des modèles de ML est leur tendance à ne pas s'accorder sur les détections : certains modèles vont qualifier une entrée comme une attaque tandis que d'autres vont la qualifier comme étant bénigne. Identifier le modèle qui trouve la bonne réponse est compliqué car les modèles de ML sont souvent considérés comme des boîtes noires et n'apportent pas d'arguments pour justifier leurs résultats. L'apprentissage ensembliste [1], [2] est une approche qui combine plusieurs modèles d'apprentissage automatique pour obtenir un système plus performant. En particulier, l'empilement de modèles consiste à combiner plusieurs modèles de base qui effectuent la même tâche (e.g., vote majoritaire pondéré). Une méthode d'empilement plus sophistiquée est le méta-apprentissage, où les sorties de plusieurs modèles de base servent d'entrée à un modèle de niveau supérieur, appelé méta-modèle. Nous proposons une nouvelle approche ensembliste qui présente systématiquement les résultats de chaque modèle de base aux analystes de sécurité, de façon à faciliter une prise de décision éclairée. Cette décision serait prise grâce à une combinaison de représentations visuelles concises des anomalies réseau et d'un niveau élevé d'explicabilité de chaque modèle de base.

L'explicabilité [3], [4] est la propriété d'un système qui rend son raisonnement et ses résultats compréhensibles par des humains. Dans les NIDSs actuels, les analystes de sécurité prennent des décisions pour résoudre les problèmes de sécurité en se basant sur l'analyse du système. Par conséquent, fournir des preuves intelligibles des résultats des modèles d'apprentissage automatique est crucial pour que les analystes fassent confiance au système. L'explicabilité n'est pas seulement importante pour améliorer la collaboration entre les analystes de sécurité et les systèmes à base d'intelligence artificielle (IA), mais aussi pour assister les ingénieurs et les chercheurs dans la conception de systèmes plus performants, en les aidant à comprendre où et pourquoi un modèle fait des erreurs.

Nous proposons une méthode qui permet aux utilisateurs de tirer parti d'un apprentissage ensembliste pour simultanément améliorer la performance de la détection et visualiser les résultats de tous les modèles de base afin de mieux comprendre comment ces détections sont effectuées. Nous introduisons une représentation visuelle des anomalies levées par des détecteurs non supervisés (UL) au fil du temps pour à la fois aider les analystes de sécurité à comprendre ce qu'il se passe sur le réseau et permettre à notre méta-modèle, un réseau de neurones convolutif (CNN), d'identifier des schémas d'attaque [5], [6]. Notre système est censé préserver les propriétés de l'apprentissage non supervisé, y compris la détection des attaques inconnues, car la couche supervisée, le méta-modèle, analyse les sorties des modèles de base, qui sont des méta-données et non des données réseau brutes.

Nous introduisons ainsi un système, conçu pour être explicable, qui analyse et combine un ensemble de modèles non supervisés pour détecter des attaques réseau. Nos contributions sont les suivantes :

1) Un NIDS à base d'IA plus transparent, dont les résultats sont détaillés,
2) Des représentations visuelles des anomalies réseau interprétables par les utilisateurs,
3) Une méthode d'apprentissage ensembliste qui utilise un CNN comme méta-modèle pour combiner des modèles de base.

La structure de l'article sera organisée de la manière suivante :

— Dans la Section II, nous examinerons les travaux connexes sur l'apprentissage ensembliste, la détection d'attaques et l'IA explicable et nous les mettrons en perspective avec notre contribution.
— Dans la Section III, nous détaillerons la conception de notre système de détection d'anomaliess. Nous expliquerons en détail les différentes étapes du processus, c'est-à-dire, l'agrégation de flux réseau, l'attribution de scores d'anomalie avec les modèles de base, la

génération de représentations visuelles des anomalies, et la reconnaissance de schémas d'attaque.
— La Section IV présentera les résultats de chaque composante de notre système. Nous présenterons les performances de chaque modèle de base, ainsi que du modèle final.
— Dans la Section V, nous interpréterons les résultats de notre système et engagerons une discussion approfondie sur les limitations de notre approche. Nous aborderons également les perspectives d'amélioration et les directions futures de recherche.
— Enfin, dans la Section VI, nous conclurons l'article en récapitulant nos principales contributions, en soulignant les avantages de notre système de détection d'attaques réseau, et en proposant des pistes pour nos futures recherches.

II. Travaux connexes

Notre recherche explore des mécanismes pour reconstruire des schémas d'attaque en utilisant plusieurs techniques de détection d'anomalies non supervisées. Nos travaux abordent des problématiques liées à (1) l'apprentissage ensembliste, (2) la détection de schémas d'attaque et (3) l'IA explicable. Cette section aborde ces thèmes dans cet ordre, et positionne nos travaux par rapport à des travaux connexes.

En ce qui concerne les systèmes utilisant de l'apprentissage ensembliste pour la détection d'anomalies réseau, Vanerio et Casas [1] ont comparé plusieurs méta-modèles pour détecter des attaques. Les auteurs ont étudié des algorithmes de vote majoritaire pondéré où les poids ont été définis par rapport à la précision des modèles de base. Nous avons adopté une approche différente où le méta-modèle est une autre couche de ML, un CNN, qui prend en entrée une représentation visuelle des détections des modèles de base et détecte des scénarios d'attaque sur ces représentations. Mirsky et al. [7] ont proposé Kitsune, un ensemble d'autoencodeurs pour détecter des anomalies réseau. Ce système est basé sur des autoencodeurs, qui sont souvent considérés comme des techniques non supervisées car ils ne nécessitent pas de données labellisées. En revanche, ils nécessitent une phase d'entraînement sur un ensemble de données normales pour pouvoir caractériser le comportement normal du système. Kitsune empile des autoencodeurs en utilisant un autre autoencodeur comme méta-modèle pour traiter les scores d'anomalie générés par les autoencodeurs de base. Leur approche pour combiner des modèles de base est proche de la notre, à la différence que nous traitons des algorithmes et des représentations de données hétérogènes.

Sur un sujet différent, Zhou et al. [8] ont proposé un système qui utilise LSTM pour détecter des attaques en plusieurs étapes. Leur modèle traite les séquences générées par le NIDS Snort [9] et traite le problème des relations à long terme entre les alertes. Ghafir et al. [10] ont proposé un système de détection de menaces persistantes avancées (APT). Le système utilise un modèle de Markov caché (HMM) pour détecter le scénario d'APT le plus probable étant données les alarmes. Ensuite, il prévoit la prochaine étape de l'APT en cours. Une différence significative avec notre travail est que le système a été entraîné par rapport à un cycle de vie d'attaque donné [11], [12], alors que notre système apprend les schémas d'attaque directement à partir de la donnée. Wang et al. [6] ont converti des données de trafic brutes (fichiers pcap) en images et ont aussi identifié des schémas d'attaques.

Wei et al. [13] ont observé que les méthodes d'explications généralistes actuelles, comme SHAP [14] et LIME [3] ne sont pas exploitables pour les NIDSs car ces méthodes ne considèrent pas les dépendances entre les caractéristiques des flux réseaux. Han et al. [4] se sont orientés vers solutions pour de l'IA explicable en proposant un système pour interpréter des NIDS utilisant de l'apprentissage profond non supervisé. Ce système analyse chaque détection d'un modèle en retournant les caractéristiques les plus importantes et en décrivant leur signification pour qu'un analyste de sécurité puisse les comprendre. Plutôt que d'investiguer des explications *ad hoc*, nous avons cherché à concevoir un système plus transparent qui fournit un ensemble de représentations intermédiaires pour aider les analystes de sécurité à comprendre les réactions du système et à mieux identifier de potentielles erreurs.

III. Conception du système

A. Aperçu du système

Nous avons développé un NIDS basé sur l'IA qui utilise un ensemble de modèles de base non supervisés pour détecter des attaques réseau (Figure 1). Un aspect clé de notre étude est de combiner différents algorithmes de détection d'anomalies en les appliquant sur les mêmes données. Les termes "modèle" et "algorithme" sont utilisés de manière interchangeable pour décrire les différentes méthodes d'apprentissage automatique utilisées dans notre système. Ainsi, nous cherchons à caractériser les résultats des modèles de base de manière indépendante de la définition des algorithmes utilisés. Pour cela, nous avons pris en compte uniquement les entrées et les sorties des modèles de base pour les comparer.

En ce qui concerne les entrées des modèles, nous avions besoin d'une représentation qui soit sémantiquement intéressante pour que les modèles puissent identifier des attaques (Section III-B). De façon similaire à UNADA [15], [16], nous avons agrégé des flux réseau par adresse IP source et destination, puis défini des caractéristiques statistiques des agrégats (Tableau I).

Chaque modèle de base attribue un score d'anomalie à un agrégat. Pour cela, on applique un même algorithme sur des sous-espaces de caractéristiques différents. Ensuite, on combine leurs sorties en les additionnant [15], [16]. La définition de nos scores est donc ainsi indépendante de l'algorithme utilisé, ce qui nous permet de comparer des détecteurs d'anomalies utilisant des algorithmes différents.

Pour aller plus loin dans la caractérisation des sorties des modèles, nous avons introduit une représentation visuelle des anomalies réseau (Section III-D) qui met en évidence à la fois un aspect spatial (e.g., est-ce que les anomalies affectent ou proviennent de la même adresse IP ?) et un aspect temporel

TABLEAU I – Caractéristiques des agrégats

Caractéristique	Clé d'agrégation	Description
n_dst_ip	IPsrc	Nombre d'adresses IP de destination
n_src_ip	IPdst	Nombre d'adresses IP source
n_dst_ports	IPsrc & IPdst	Nombre de ports de destination
n_src_ports	IPsrc & IPdst	Nombre de ports source
n_fwd_pkts	IPsrc & IPdst	Nombre de paquets forward
n_bwd_pkts	IPsrc & IPdst	Nombre de paquets backward
sum_flx_dur	IPsrc & IPdst	Somme de la durée des flux
tot_flx	IPsrc & IPdst	Nombre de flux
sum_pkts_size	IPsrc & IPdst	Somme de la taille des paquets
std_pkt_size	IPsrc & IPdst	Ecart-type de la taille des paquets

(e.g., est-ce que les anomalies sont répétées ?). Pour finir, notre méta-modèle, un CNN, analyse ces représentations des anomalies réseau pour identifier des schémas d'attaque et lever des alertes.

B. Agrégation de flux réseau

Notre système calcule les caractéristiques décrites dans le Tableau I à partir d'agrégats de flux réseau. Nous avons choisi d'analyser à la fois des agrégats par adresse IP source et destination car ces perspectives pourraient être complémentaires. En effet, certaines attaques (e.g., attaques par déni de service distribuées) impliquent plusieurs adresses IP source à destination d'une même adresse IP, tandis que d'autres attaques (e.g., scans de réseau) impliquent généralement une seule adresse IP source et plusieurs destinataires. L'utilisation de ces perspectives est propice à la détection de schémas d'attaque mais est également très lisible pour les analystes de sécurité. En effet, nous utilisons un nombre réduit de caractéristiques ainsi qu'une représentation par adresses IP, ce qui permet aux analystes de sécurité d'identifier les machines ayant un comportement anormal.

De plus, pour éviter de capturer des anomalies liées à des changements brutaux, mais légitimes, du trafic (e.g., jours ouvrés, week-ends), nous avons choisi d'appliquer les algorithmes de base, non supervisés, sur de petits intervalles de temps Δ_t que nous avons fixé empiriquement à 2 minutes, plutôt que sur l'ensemble du trafic.

Enfin, afin de réduire davantage la dimensionnalité de nos données, nous calculons les agrégats uniquement sur les adresses IP internes au réseau considéré, c'est-à-dire les machines appartenant au réseau de l'entreprise. Il n'y a donc pas d'agrégats par destination pour les destinations publiques sur Internet.

C. Attribution de scores d'anomalie non supervisée

Le composant décrit Figure 2 consiste en un ensemble de détecteurs d'anomalies non supervisés qui quantifient le degré d'anomalie de chaque agrégat d'entrée. Il prend en entrée les caractéristiques des agrégats (Tableau I), par adresse IP source ou destination, durant une même période de temps Δ_T, que nous avons fixée empiriquement à 30 minutes. Le composant retourne des représentations visuelles des anomalies qui se présentent sous la forme d'un ensemble de matrices. Chaque matrice contient les scores d'anomalie de chaque agrégat attribués par les différents détecteurs d'anomalies.

Afin d'être capable de détecter de nouvelles attaques, nous avons utilisé des détecteurs d'anomalies non supervisés comme modèles de base. Nous avons étudié plusieurs algorithmes de détection d'anomalies non supervisés implémentés dans scikit-learn [17] et PyOD [18] que nous avons appliqués sur des agrégats générés à partir du jeu de données CIC-CSE-IDS2018 [19]. Nous avons observé des résultats différents générés avec ces algorithmes de détection d'anomalies pour de mêmes agrégats d'entrée (Section IV-C).

Nous avons sélectionné un ensemble d'algorithmes de détection d'anomalies de façon à maximiser la détection d'attaques tout en minimisant les fausses alarmes (c'est-à-dire la détection erronée de trafic légitime comme étant une attaque). Nous avons défini des modèles de base en utilisant à chaque fois les mêmes données d'entrée, mais avec des algorithmes de détection d'anomalies différents. Ensuite, nous avons considéré un modèle *empilé* qui cumule les alarmes des modèles de base en effectuant la somme booléenne de leurs résultats. Nous avons étudié des algorithmes de base très différents pour construire notre modèle empilé, et ce afin de maximiser leur complémentarité :

- *Isolation Forest (IF)* [20] est un algorithme basé sur des arbres.
- *Local Outlier Factor* [21] est un algorithme basé sur une notion de densité.
- *One-Class SVM (OCSVM)* [22] utilise un hyperplan.
- *KNN non supervisé* [23] est un algorithme basé sur une notion de distance.
- *COPOD* [24] est une algorithme probabiliste.

Nous avons évalué la performance de toutes les combinaisons possibles de trois algorithmes parmi les algorithmes étudiés sur le jeu de données CIC-CSE-IDS2018 [19], et avons sélectionné la combinaison qui a le plus grand taux de vrais positifs (TPR) et avec le plus petit taux de faux positifs (FPR).

Pour caractériser et comparer les résultats des modèles de base, que nous avons obtenus avec ces différents algorithmes non supervisés, nous avons utilisé une quantification des anomalies indépendante des algorithmes, définie précédemment dans UNADA [15], [16]. Pour attribuer un score d'anomalie avec un modèle de base, nous avons considéré tous les sous-espaces de $k = 2$ caractéristiques parmi les $n = 9$ caractéristiques des agrégats et avons appliqué un même algorithme de détection d'anomalies sur chaque sous-espace. Le score d'anomalie d'un agrégat généré par un modèle de base est défini comme le nombre de fois qu'un agrégat a été détecté comme anormal. Ces scores vont de 0, pour un agrégat complètement normal (d'après le modèle), à 36 pour un agrégat complètement anormal, car cela correspond au nombre de combinaisons de $k = 2$ caractéristiques parmi $n = 9$:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

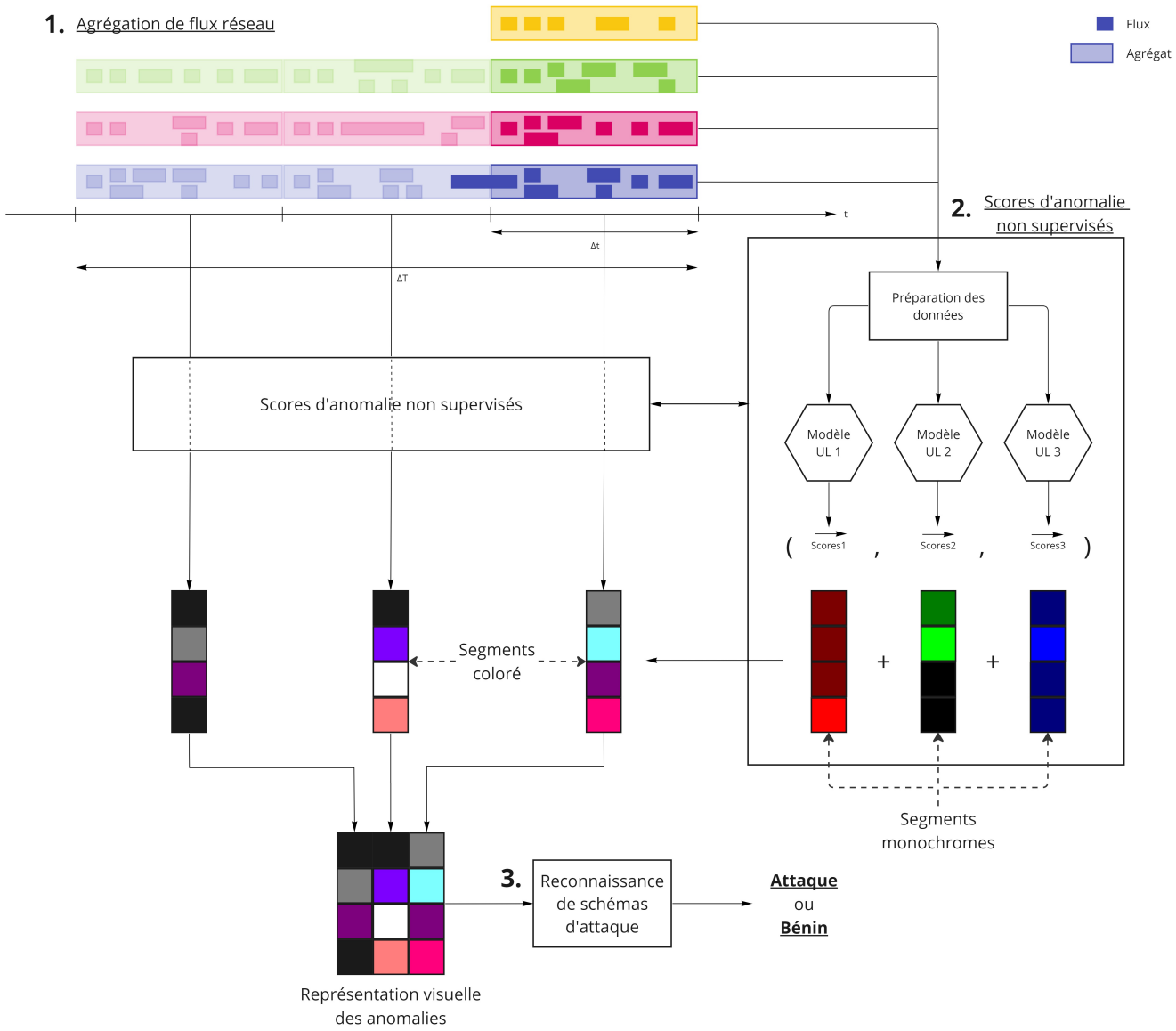


FIGURE 1 – Aperçu du système. Le système analyse une capture de trafic pendant une période Δ_T. Tout d'abord, il agrège les flux réseaux de la capture sur des intervalles de temps Δ_t et calcule leurs caractéristiques. Ensuite, il applique un ensemble de $N = 3$ modèles de base non supervisés pour attribuer des scores d'anomalie aux agrégats obtenus. Après cela, le système génère une représentation visuelle des anomalies détectées, permettant une visualisation claire des schémas d'attaque potentiellement présents dans la capture. Enfin, ces représentations sont analysées par un module de reconnaissance de schémas d'attaque, qui détermine si le réseau est attaqué pendant la période Δ_T.

D. Reconnaissance de schémas d'attaque

Dans la section précédente, nous avons sélectionné un ensemble de modèles non supervisés pour détecter des agrégats anormaux. Nous avons introduit une dimension temporelle à notre représentation des anomalies (Figure 4) pour mettre en évidence les séquences d'anomalies détectées par chaque modèle de base pendant une période de temps Δ_T de 30 minutes. Nous avons représenté les sorties de chaque modèle de base comme une matrice de scores d'anomalie où chaque ligne correspond à une adresse IP, source ou destination selon la clé d'agrégation, et chaque colonne correspond à un intervalle de temps.

Nous pouvons analyser ces représentations en utilisant des modèles d'apprentissage profonds, tels que des CNNs. Nous utilisons ainsi un CNN comme méta-modèle pour identifier des schémas d'attaque sur ces représentations du trafic. Le CNN prend en entrée les deux ensembles de matrices, qui représentent les anomalies générées à partir des sorties des modèles de base sur les agrégats par adresse IP source et destination pendant une période de temps Δ_T. Le CNN donne

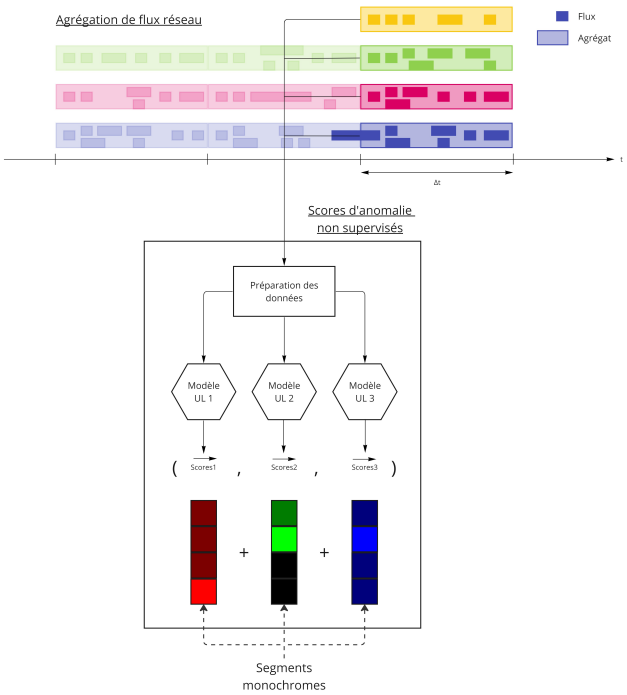


FIGURE 2 – Attribution de scores d'anomalie par 3 modèles de base non supervisés pendant un intervalle de temps Δ_t. Cette analyse génère un segment vertical qui fait partie de la représentation visuelle des anomalies générée dans la Figure 3. Les flux réseau sont regroupés en agrégats (4 dans cet exemple) pendant un intervalle de temps Δ_t. Ensuite, les caractéristiques des agrégats sont extraites et préparées avant d'être évaluées par N modèles de base non supervisés ($N = 3$ dans cet exemple). Chaque modèle de base attribue un score d'anomalie à chaque agrégat. Une couleur est assignée à chaque modèle, ce qui permet de représenter le scores d'anomalie sur un segment monochrome. En superposant les segments monochromes des N modèles de base, nous obtenons un segment coloré qui représente les anomalies détectées par l'ensemble des modèles pendant l'intervalle de temps Δ_t.

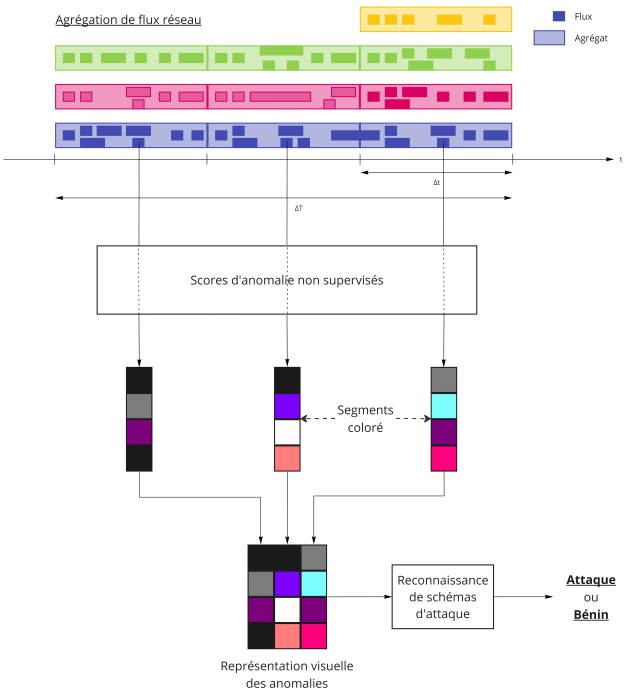


FIGURE 3 – Reconnaissance de schémas d'attaque sur une représentation visuelle des anomalies détectées pendant une période de temps Δ_T. La période Δ_T est constituée, dans cet exemple, de 3 intervalles de temps consécutifs Δ_t. Les flux réseau de chaque intervalle de temps Δ_t sont analysés comme dans la Figure 2. Chaque intervalle de temps Δ_t est représenté par un segment coloré. En disposant les 3 segments côte à côte, nous obtenons une représentation visuelle des anomalies réseau détectées par les $N = 3$ modèles de base pendant la période Δ_T.

le résultat final du système de détection, en indiquant si le réseau est attaqué durant la période Δ_T.

Le CNN nécessite un jeu de données labellisé pour son entraînement et sa validation. Nous avons choisi de labelliser une représentation comme étant une attaque si elle contient au moins un agrégat d'attaque (ce qui veut dire que le réseau subit une attaque pendant la capture), et bénigne sinon.

Pour visualiser les analyses de chaque modèle de base, nous leur avons assigné une couleur (rouge pour Local Outlier Factor, bleu pour KNN, et vert pour COPOD). L'intensité de la couleur est proportionnelle au score d'anomalie généré par le modèle. Ainsi, un pixel blanc signifie que tous les modèles ont détecté l'agrégat comme très anormal et un pixel noir signifie qu'aucun des modèles n'a levé une alarme (Figure 4).

IV. RÉSULTATS

A. *Jeu de données*

Nous avons évalué notre système sur le jeu de données CIC-CSE-IDS2018 [19]. Sharafaldin et al. [19] ont implémenté un réseau d'entreprise réaliste avec 420 machines et 30 serveurs, et une infrastructure d'attaque constituée de 50 machines. Le jeu de données consiste en dix jours de captures réseau pendant les horaires de travail. Les auteurs ont simulé plusieurs scénarios d'attaque, notamment des attaques par force brute, des attaques par déni de service distribuées, des injections SQL, etc. Le jeu de données fournit les heures des attaques ainsi que les adresses IP de l'attaquant et des victimes, nous avons donc labellisé les flux réseau en conséquence. Un agrégat par source ou par destination est labellisé comme une attaque si la source ou la destination émet ou reçoit du trafic depuis ou vers une adresse IP attaquante. Finalement, une image est labellisée comme une attaque si un agrégat dans l'image est étiqueté comme une attaque.

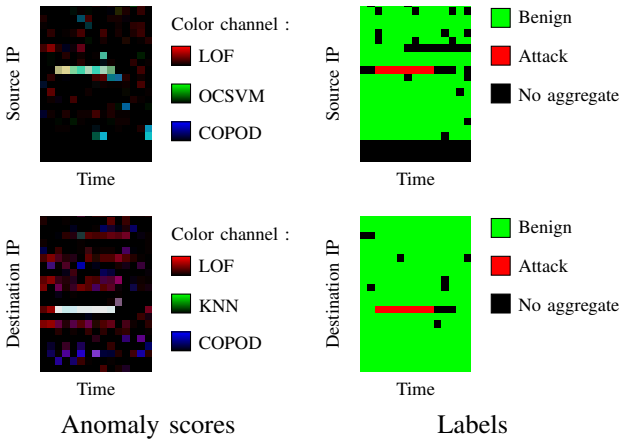


FIGURE 4 – Représentation du trafic pendant une attaque par déni de service. Les images à droite affichent les labels, où chaque pixel rouge représente un agrégat d'attaque et chaque pixel vert représente un agrégat bénin. Nous observons une ligne rouge sur chacune des images, ce qui signifie que l'attaque provient et touche une seule IP et que les agrégats sont rapprochés dans le temps. Les images à gauche ont été générées en utilisant les modèles décrits Section III-C. Sur les images de gauche, nous observons que les ensembles de modèles non supervisés ont détecté la ligne d'attaque entière.

B. Agrégation de flux réseau

Notre système agrège d'abord les flux réseaux par adresse IP source et destination (Section III-B). Le Tableau II décrit la répartition des agrégats ainsi obtenus pour chaque catégorie de trafic. On constate qu'il y a moins d'agrégats par IP destination que par IP source, ce qui indique qu'il y a davantage de machines émettrices de trafic que de machines réceptrices dans cette capture de trafic.

C. Attribution de scores d'anomalie non supervisée

Notre système génère ensuite des représentations intermédiaires visuelles des anomalies réseau (Figure 4). Pour cela, nous avons sélectionné un ensemble d'algorithmes qui génèrent des résultats complémentaires (Section III-C). Le tableau III montre la performance des cinq meilleures combinaisons de modèles de base, que l'on appelle *modèles empilés*, sur les agrégats par adresse IP source et destination. Il indique leurs taux de vrais positifs (TPR) et de faux positifs (TFP) sur les agrégats par source et destination. Nous considérons que le modèle empilé détecte un agrégat comme une attaque si l'un des modèles de base détecte l'agrégat comme une attaque.

Les meilleures combinaisons sont (LOF, OCSVM, COPOD) pour analyser les agrégats par adresse IP source, et (LOF, KNN, COPOD) pour analyser les agrégats par adresse IP de destination. Bien que nous ayons sélectionné la combinaison avec les meilleurs résultats de manière empirique, notre intuition est que cette performance provient de l'utilisation d'algorithmes avec des méthodes très différentes pour détecter les anomalies.

Ensuite, nous avons appliqué ces différents algorithmes sur chaque sous-espace de $k = 2$ caractéristiques des agrégats d'un même intervalle de temps Δ_t et cumulé les alarmes levées sur chaque sous-espace pour définir un score d'anomalie associé à chaque modèle de base.

Le Tableau IV décrit les taux d'attaques détectées, ou vrais positifs (TVP) et de fausses alarmes, ou faux positifs (TFP) pour chaque modèle de base ainsi que pour le modèle empilé, obtenu en cumulant les alarmes levées par chacun des modèles de base. On observe que le modèle le plus performant en terme de TVP ne détecte pas certaines attaques que des modèles moins performants ont détectées. Le modèle empilé a un meilleur TVP mais lève plus de fausses alarmes. Le traitement de l'ensemble des alarmes levées par ce modèle empilé sera effectué par le méta-modèle Section IV-D.

D. Reconnaissance de schémas d'attaque

Dans la section précédente, nous avons attribué un score d'anomalie par modèle à chaque agrégat. Cette étape nous permet de générer un segment monochrome par modèle de base et par intervalle de temps. Nous superposons ensuite les segments de chaque couleur correspondant au même intervalle de temps Δ_t et les disposons côte à côte pour représenter le trafic sur une période de temps plus longue Δ_T.

Enfin, nous avons obtenu un jeu de données de 4214 paires d'images, qui représentent les scores d'anomalie des agrégats sur une période Δ_T. Ce jeu de données sera utilisé pour entraîner et évaluer le module de reconnaissance de schémas d'attaque (Section III-D).

Le Tableau V montre le F-score et la matrice de confusion de notre module de reconnaissance de schémas d'attaque, i.e., notre CNN, qui analyse des représentations d'agrégats par adresse IP source et destination (le modèle combiné). Pour déterminer si les deux représentations sont complémentaires pour le modèle, nous avons comparé la performance du modèle combiné avec celle des modèles qui traitent uniquement les représentations des agrégats avec une seule clé d'agrégation (adresse IP source ou destination). Nous observons dans le Tableau V que le modèle combiné a un meilleurs F-score que les deux autres modèles, il n'a levé aucun faux positif, cependant, il a détecté moins d'attaques réelles que le modèle qui ne regarde que les adresses IP de destination.

V. DISCUSSION

Nous avons proposé un système qui combine plusieurs détecteurs d'anomalies non supervisés en utilisant un CNN sur une représentation visuelle du trafic. Cette méthode de combinaison a permis d'améliorer significativement la performance de détection du système. Ce système génère plusieurs représentations intermédiaires qui peuvent aider à identifier les erreurs restantes.

Dans cet article, nous avons détecté des anomalies sur des agrégats par adresse IP source et destination pendant des intervalles de temps de 2 minutes. Cependant, ce niveau de granularité peut ne pas être adapté pour détecter certaines attaques. Par exemple, sur la Figure 4, on observe que nos

TABLEAU II – Répartition des agrégats par adresse IP source et destination pour chaque catégorie de trafic

	IP source		IP de destination	
Catégorie de trafic	**Nombre**	**Proportion (%)**	**Nombre**	**Proportion (%)**
Attaque par force brute	41	0.004455	97	0.009547
Déni de service	13	0.001412	70	0.004231
Attaque web	0	0	133	0.013089
Infiltration	32	0.003477	76	0.007480
Bot	1500	0.162978	0	0
Déni de service distribué	58	0.006302	62	0.006102
Bénin	918723	99.821376	1015651	99.956894
Total	**6516**	**100**	**3486**	**100**

TABLEAU III – 5 meilleures combinaisons de 3 modèles de base pour les agrégats par source et destination. Les combinaisons sont classées selon deux critères : le TPR, représentant le pourcentage d'attaques correctement détectées, et le TFP, indiquant la proportion de fausses alarmes.

IP source			IP de destination		
Sous-ensemble de modèles de base	**TVP**	**TFP**	**Sous-ensemble de modèles de base**	**TVP**	**TFP**
(LOF, OCSVM, COPOD)	0.9878	0.6763	(LOF, KNN, COPOD)	0.9384	0.3994
(IF, LOF, OCSVM)	0.9811	0.6704	(LOF, OCSVM, COPOD)	0.9384	0.4437
(LOF, OCSVM, KNN)	0.9732	0.6762	(LOF, OCSVM, KNN)	0.9315	0.4312
(LOF, DBSCAN, OCSVM)	0.9726	0.6620	(LOF, DBSCAN, COPOD)	0.9292	0.3917
(IF, OCSVM, COPOD)	0.9690	0.4937	(IF, LOF, COPOD)	0.9292	0.3939

TABLEAU IV – Taux d'attaques détectées (TVP) et de fausses alarmes (TFP) pour les modèles de base

IP source			IP de destination		
Modèle de base	**TVP**	**TFP**	**Modèle de base**	**TVP**	**TFP**
LOF	0.8923	0.3754	LOF	0.9041	0.3366
OCSVM	0.8394	0.4634	KNN	0.8904	0.1862
COPOD	0.7652	0.1467	COPOD	0.8744	0.1795
Modèle empilé	**0.9878**	**0.6763**	**Modèle empilé**	**0.9384**	**0.3994**

TABLEAU V – Résultats des CNNs

	TVP	**TFP**	**F-score**	**Matrice de confusion** [**VP FP** / **VN TN**]
CNN Source	0.9796	0.0062	0.9905	[96 2 / 2 322]
CNN Destination	0.9796	0.0093	0.9882	[96 3 / 2 321]
Combined CNN	**0.9898**	**0.0093**	**0.9906**	[**97 3** / **1 321**]

détecteurs d'anomalies n'ont pas détecté la totalité de l'attaque par force brute sur les agrégats par IP de destination. Cela peut s'expliquer par le fait que la machine victime reçoit des paquets malveillants agrégés avec des paquets légitimes, ce qui rend cette attaque moins identifiable au niveau des agrégats par adresse IP de destination.

De plus, nous avons sélectionné la combinaison de modèles de base qui détecte le plus grand nombre d'attaques tout en minimisant les fausses alarmes. Cependant, il est important de souligner que cette combinaison est statique et que le critère de sélection des modèles est conservateur, ce qui peut entraîner un grand nombre de fausses alarmes dans les résultats. Dans certains cas très spécifiques, un détecteur ayant une performance inférieure peut être le seul à détecter correctement une attaque. Il pourrait être intéressant d'explorer une sélection dynamique des modèles, qui interrogerait ce détecteur uniquement lorsque cela est nécessaire pour obtenir une détection précise.

Pour finir, nous avons observé la présence d'un à la fin d'une attaque. Notre système pourrait bénéficier des techniques d'analyse de vidéos pour détecter les attaques dans leur intégralité.

Dans nos travaux futurs, nous évaluerons notre système sur un jeu de données plus volumineux et évaluerons sa capacité à détecter de nouvelles attaques.

VI. CONCLUSION

Nous avons proposé un système de détection d'attaques réseau conçu pour être explicable. Dans notre approche, nous avons utilisé des techniques non supervisées pour détecter des anomalies sur des agrégats basés sur les adresses IP source et destination. Les résultats produits par notre système sont interprétables et compréhensibles pour les analystes de sécurité. Les sorties incluent les adresses IP associées aux agrégats identifiés comme anormaux, ce qui permet aux analystes de les relier à des machines spécifiques dans le

réseau. De plus, les couples de caractéristiques anormaux sont également rapportés, ce qui fournit ainsi un niveau de détail supplémentaire sur les anomalies détectées. Pour représenter ces anomalies, nous avons utilisé des images générées à partir d'un ensemble de modèles de base non supervisés. Ces images permettent aux analystes de sécurité d'observer visuellement les schémas d'attaque identifiés par le système. Enfin, nous avons analysé ces représentations du trafic en utilisant un CNN pour détecter des schémas d'attaque.

Notre approche vise donc à concevoir un système de détection d'attaques plus transparent, qui permet aux analystes de sécurité de suivre les décisions prises par le système. L'évaluation de notre système sur une partie du jeu de données CIC-CSE-IDS2018 [19] a démontré une précision correcte, mais surtout, les erreurs commises par le système sont facilement identifiables et peuvent être analysées par les analystes de sécurité.

Une perspective prometteuse est la détection de nouvelles attaques. Comme les attaquants développent constamment de nouvelles attaques, il est essentiel de développer des systèmes capables de détecter et de caractériser ces attaques inconnues. Dans notre système, nous avons utilisé des modèles non supervisés pour détecter les anomalies. Il serait désormais intéressant d'évaluer la capacité de notre système à identifier des schémas d'attaques inconnus.

RÉFÉRENCES

[1] J. Vanerio and P. Casas, "Ensemble-learning Approaches for Network Security and Anomaly Detection," in *Proceedings of the Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*. Los Angeles CA USA : ACM, Aug. 2017, pp. 1–6.

[2] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An Adaptive Ensemble Machine Learning Model for Intrusion Detection," *IEEE Access*, vol. 7, pp. 82 512–82 521, 2019.

[3] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why Should I Trust You ?" : Explaining the Predictions of Any Classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA : Association for Computing Machinery, Aug. 2016, pp. 1135–1144.

[4] D. Han, Z. Wang, W. Chen, Y. Zhong, S. Wang, H. Zhang, J. Yang, X. Shi, and X. Yin, "DeepAID : Interpreting and Improving Deep Learning-based Anomaly Detection in Security Applications," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. Virtual Event Republic of Korea : ACM, Nov. 2021, pp. 3197–3217.

[5] I. Ghafir, M. Hammoudeh, V. Prenosil, L. Han, R. Hegarty, K. Rabie, and F. J. Aparicio-Navarro, "Detection of advanced persistent threat using machine-learning correlation analysis," *Future Generation Computer Systems*, vol. 89, pp. 349–359, Dec. 2018.

[6] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *2017 International Conference on Information Networking (ICOIN)*, Jan. 2017, pp. 712–717.

[7] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune : An Ensemble of Autoencoders for Online Network Intrusion Detection," in *Proceedings 2018 Network and Distributed System Security Symposium*. San Diego, CA : Internet Society, Feb. 2018.

[8] P. Zhou, G. Zhou, D. Wu, and M. Fei, "Detecting multi-stage attacks using sequence-to-sequence model," *Computers & Security*, vol. 105, p. 102203, Jun. 2021.

[9] "Snort - Network Intrusion Detection & Prevention System," https ://www.snort.org/.

[10] I. Ghafir, K. G. Kyriakopoulos, S. Lambotharan, F. J. Aparicio-Navarro, B. Assadhan, H. Binsalleeh, and D. M. Diab, "Hidden Markov Models and Alert Correlations for the Prediction of Advanced Persistent Threats," *IEEE Access*, vol. 7, pp. 99 508–99 520, 2019.

[11] E. Hutchins, M. Cloppert, and R. Amin, "Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains," in *ICIW2011-Proceedings of the 6th International Conference on Information Warfare and Secuirty : ICIW*. Academic Conferences Limited, 2011, pp. 113–125.

[12] D. McWhorter, "Mandiant Exposes APT1 — One of China's Cyber Espionage Units & Releases 3,000 Indicators," *Mandiant, February*, 2013.

[13] F. Wei, H. Li, Z. Zhao, and H. Hu, "xNIDS : Explaining deep learning-based network intrusion detection systems for active intrusion responses," in *32nd USENIX Security Symposium (USENIX Security 23)*. Anaheim, CA : USENIX Association, Aug. 2023, pp. 4337–4354.

[14] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *Advances in Neural Information Processing Systems*, vol. 30. Curran Associates, Inc., 2017.

[15] P. Casas, J. Mazel, and P. Owezarski, "UNADA : Unsupervised Network Anomaly Detection Using Sub-space Outliers Ranking," in *10th IFIP Networking Conference (NETWORKING)*, vol. LNCS-6640. Springer, May 2011, pp. 40–51.

[16] J. Dromard, G. Roudière, and P. Owezarski, "Online and Scalable Unsupervised Network Anomaly Detection Method," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 34–47, Mar. 2017.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, and D. Cournapeau, "Scikit-learn : Machine Learning in Python," *MACHINE LEARNING IN PYTHON*, p. 6, 2011.

[18] Y. Zhao, Z. Nasrullah, and Z. Li, "PyOD : A Python Toolbox for Scalable Outlier Detection," *Journal of Machine Learning Research*, vol. 20, no. 96, pp. 1–7, 2019.

[19] I. Sharafaldin, A. Habibi Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization :," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy*. Funchal, Madeira, Portugal : SCITEPRESS - Science and Technology Publications, 2018, pp. 108–116.

[20] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in *2008 Eighth IEEE International Conference on Data Mining*, Dec. 2008, pp. 413–422.

[21] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF : Identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA : Association for Computing Machinery, May 2000, pp. 93–104.

[22] B. Schölkopf, R. C. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support Vector Method for Novelty Detection," in *Advances in Neural Information Processing Systems*, vol. 12. MIT Press, 1999.

[23] F. Angiulli and C. Pizzuti, "Fast Outlier Detection in High Dimensional Spaces," in *European conference on principles of data mining and knowledge discovery*, T. Elomaa, H. Mannila, and H. Toivonen, Eds. Berlin, Heidelberg : Springer, 2002, pp. 15–27.

[24] Z. Li, Y. Zhao, N. Botta, C. Ionescu, and X. Hu, "COPOD : Copula-Based Outlier Detection," in *2020 IEEE international conference on data mining (ICDM)*. IEEE, Nov. 2020.

Trust in Automation: Analysis and Model of Operator Trust in Decision Aid AI Over Time

Vincent Fer
Thales DMS & IMT Atlantique
Brest, France
vincent.fer@imt-atlantique.fr

Daniel Lafond
Thales Research and Technology
Quebec, Canada
daniel.lafond@thalesgroup.com

Gilles Coppin
IMT Atlantique
Brest, France
gilles.coppin@imt-atlantique.fr

Mathias Bollaert
Thales Defence Mission Systems
Brest, France
mathias.bollaert@fr.thalesgroup.com

Olivier Grisvard
Thales Defence Mission Systems
Brest, France
olivier.grisvard@thalesgroup.com

Pierre De Loor
ENIB
Brest, France
pierre.deloor@enib.fr

***Abstract*—Understanding how human trust in AI evolves over time is essential to identify the limits of each party and provide solutions for optimal collaboration. With this goal in mind, we examine the factors that directly or indirectly influence trust, whether they come from humans, AI, or the environment. We then propose a summary of methods for measuring trust, whether subjective or objective, to show which ones are best suited for longitudinal studies. We then focus on the main driving force behind the evolution of trust: feedback. We justify how learning feedback can be transposed to trust and what types of feedback can be applied to impact the evolution of trust over time. After understanding the factors that influence and how to measure trust, we propose an application example on a maritime surveillance tool with an AI-based decision aid.**

***Index Terms*—Trust in Automation, Maritime Patrol, Longitudinal Experiment, Feedback, Human Factors, Cognitive Engineering**

I. Introduction

With the emergence of AI several years ago in the world of research and then a sudden acceleration in its accessibility to a large part of the population in the civilian or military sectors, it is becoming essential to determine the level of trust that a human should be able to place in these intelligent systems. Today, used in the medical world to detect disease, in the banking sector to detect fraud, in IT to develop ultra-fast applications, or in the military to assist operators in high-risk operations, AI is becoming a central element in boosting performance. This integration is not without consequences, since humans tend to become over- or under-confident in automation [1] over time, this can lead to catastrophic results such as fatal accidents [2]. Confidence can generally be distinguished from trust in the following sense: confidence is defined as the expectation of a certain level of performance, while trust is a human attitude based on the perception of an agent's ability to help him perform a task in a situation characterized by uncertainty and vulnerability [3]. Trust is obviously not unique to the Human-AI pairing, but is present in all areas of our society, whether it is trust in our institutions, trust in science, trust in business, trust in justice, or trust in others. The same is true for research with a wide variety of disciplines that have been working on this subject for many years, including sociology, psychology, philosophy, neurology, informatics, etc. What is interesting about this diversity is that it is becoming clear that trust cannot be dealt with by a single discipline. This is what Lee & See have attempted to do with the transposition of concepts related to Human-Human trust to Human-Automation trust. A model in which trust is ultimately just one state in a cycle of cognitive evolution in humans has been proposed. The current question is not "Is the human trustworthy?" but "Is AI trustworthy?" For this first question, which may be essential in mission-critical systems, studies are underway, such as Hou's [4] article on the integration of decision-support AI, in which the AI can make a decision when the Human is no longer in a position to do so, in order to complete a mission. We propose to address the second question in the following sections, with a focus in Section 2 on trust in automation, its influencing factors, and how to measure it. In Section 3, we transpose learning feedback to feedback as a driver of trust evolution. In Section 4, we define a synthetic model based on models in the literature and enhanced by the notion of feedback. In the final section, Section 5, we propose an example of an application for maritime surveillance. Finally, we conclude this paper by summarizing our proposals and talking about our future experiments.

II. State of the art

A. Trust

Trust, a term at the core of Human-AI cooperation, is a major issue to be understood in the future to optimize Human and AI performance in rich and varied contexts. Humans are strong in contexts requiring nuance, while AI can handle large amounts of generic data. We explore different processes impacting trust that have a unique influence and occur at different temporalities. We will also show that the decision to trust is not solely impacted by trust but can also be influenced by factors specific to each context. To understand what trust is, let us go back to Lee & See's [3] definition, which is "trust is

a human attitude based on the perception of an agent's ability to help them perform a task in a situation characterized by uncertainty and vulnerability" [3]. In the context of human-autonomy teams, this definition highlights the three groups of factors that influence trust: trustor factors (human), trustee factors (AI), and environmental factors. [5]–[7].
These three sets of factors can be described as follows:

- Trustor factors: These factors are related to human operators, including their personality traits, previous experience with AI systems, cognitive abilities, and emotional states [8]. For the same experience with an AI, an operator with specific features of personality [9] - such as extroversion, for instance - can reveal to be more trustful than someone with another personality profile.
- Trustee factors: These factors relate to the AI system itself, such as its performance [10], process [11], transparency [12], and explainability [13]. An AI system that continuously provides accurate recommendations and is easy to understand will gain more trust from the operator. Factors such as system errors, poor communication, or lack of transparency can negatively impact trust [14].
- Environmental factors: These factors include the context and conditions under which the Human-AI interaction occurs. Factors such as task complexity, time pressure [15], or situational risk [16] can influence the operator's trust or trust-related behavior toward the AI system. For example, in high-risk environments, operators will be more cautious when relying on the AI system, especially if the potential consequences of an error are severe [16], [17].

Although influenced by different actors, trust is part of a cognitive cycle. Lee & See [3] propose a general model for the evolution of trust, including beliefs, attitudes, intentions, and behaviors. We synthesize this model in Fig.1. Beliefs are initially formed with "external" elements, such as rumors, training, or knowledge of a similar system, and will evolve over the course of the user's experience. These latest influence the user's attitude which also depends on characteristics specific to each individual (culture, predisposition to trust). Trust will have an impact on an individual's intention to use the system or not. Factors related to the context (time pressure) or the individual (perceived risk [16], workload [18]) can also influence this intention. Intention is not directly observable and may only be expressed through a behavior that is concrete evidence of whether or not an individual trusts the system. The most classical examples [19] of such behaviors are reliance, compliance, and verification time. Reliance consists of asking the system for a recommendation. Compliance consists of changing one's decision in response to a system proposal. The characteristics of behavior, such as verification time, are also meaningful: The shorter the time required to verify a recommendation, the higher the level of trust. However, intention can be influenced by many other factors, so observed behaviors cannot be directly mapped onto trust itself. One such example is the time pressure that can be encountered in different operational contexts. Time pressure arises when individuals face limited time to perform tasks. In such instances, the integration of AI becomes an important aid in improving performance while reducing the risk of critical errors. Several benefits can be observed, such as reduced workload, reduced stress, and increased performance. However, the Rieger article [20] confirms the findings of Rice and Keller [15]: time pressure can result in excessive reliance and an escalation of errors. Nonetheless, Rice and Keller [15] demonstrates that under high time pressure, mitigate reliance on AI can enhance overall performance compared to an operator working alone.

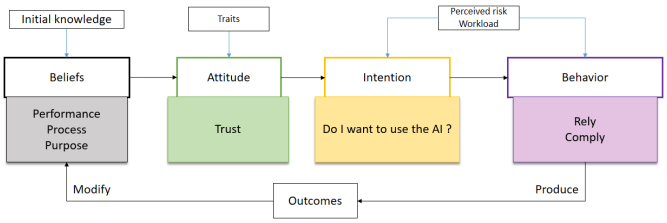


Fig. 1. The simplified Lee & See [3] model describing the evolution of trust in a cognitive cycle. Blue arrows represent influences outside the cycle, while black arrows represent evolution within the cycle.

To measure human trust in AI over its evolution at different moments, there are various subjective and objective methods [19], each with its own advantages and disadvantages. Among the subjective ones, self-report measures of trust, such as the Checklist for Trust [21] or the Measures of Trust & Trustworthiness [22] are easy to administer, but may be intrusive when used during experimentation. It can be used before or after an experiment to capture the attitude of trust and the beliefs. Behavioral and physiological measures, such as electroencephalography (EEG) [23], [24], provide more objective data but can be influenced by factors not related to trust and can require specialized equipment. It may be used during the experiment and will give online data about the behaviors but also on the attitude and intention. Researchers often use a combination of these methods to obtain a complete understanding of the dynamics of trust [1], [19], [25]. Currently, there is no general computational model of the evolution of trust, although some attempt to predict how trust evolves based on controlled variables limited to a specific context and with limited precision [**?**], [26], [27]. The gap lies in the fact that each context has unique variables with various impacts on the level of individuals trust. They are trying to estimate the level of trust at each point in time. Each event perceived by the system will change the value of trust or another variable. Estimating trust is important for regulating it. If a computational model is able to detect that an operator is becoming over- or under-confident towards the AI, then methods can be put in place to adjust it to the expected level before it is reached. General methods are proposed, for example, by de Visser [14] with apologies, explanation of repair, expression of system limits, or regular alerts for trust reduction. Certain methods are not applicable to all systems, and the appropriate ones must be selected for each context.

While there is currently no exact answer to the question

of how trust evolves, studies have highlighted the main factors that have an impact on trust [3], [5], [8], such as AI performance, its process, or its initial purpose. The impact of AI performance on trust has been demonstrated in numerous studies [5], [6], [10]. Few studies have been conducted to date on the process [28] or purpose [7].

- Performance is easily identifiable by comparing an expected result with the result obtained, or the speed with which a task is completed, with or without AI. Three metrics of performance are observable: Human performance (without AI), AI performance (without human) and team performance (Human + AI). Avril [10] studies the impact of AI performance on trust, and, indeed, trust increased when AI performance was high (87. 5%) compared to low (67.5%). Other information can be observed, such as the more information the AI gives about its choices, the better the Human-AI pairing performs. A change in behavior can also be observed when the information about a case is very precise and when no information is provided. No differences were observed when there was little or no information about potential errors. However, humans tend to see fewer AI errors with a high-performance AI (87.5%) than with a low-performance AI (67.5%). No correlation was found between self-confidence and AI performance.
- Concerning the process, which corresponds to the way the AI works or the steps it takes, this is not necessarily visible to the operator, and this is what studies on explainability and transparency are trying to resolve, to shed light on what are known as "black boxes".
- Purpose refers to how the AI is used, and more specifically how it is used in relation to its initial task. The operator perceives whether the AI's purpose is to help him in the task at hand, or whether it has not been developed to perform this task. A difference between the developers' initial goal and the goal expected by users can create disillusionment, which can reduce trust in automation. It is therefore important to communicate the latter's limits, with the goal of optimal collaboration.

Particular difficulties occur when trying to apply these kinds of indicators for the long-term analysis of operator trust, which is still a largely unexplored sub-field of research [29]. Among the rare references related to this topic, one can mention Beggiato's [11] study, which lasted two months. The research examined the evolution of trust and acceptance as drivers learned about the system. The results showed that trust and acceptance increased with familiarity, emphasizing the importance of learning in shaping users relationships with automation.

Our assumption, given Beggiato's findings on the similarities between the learning curve and the trust evolution curve, and Lee and See's model describing beliefs (perceived performance, process, and purpose) as the antecedents of trust, is that feedback, which is essential to learning, can be transposed to trust, since beliefs evolve with experience. Based on this premise, we turned to Bosc Miné's [30] article on feedback in a learning context. It describes all forms of feedback and allows us to target the ideal feedback for each context and need. To learn, feedback is necessary and essential for each learner (student, professional, expert, etc.). Using Lee & See's simplified model, behavior is linked to an outcome. This outcome is caused by the decision to trust or not. It can either agree with what was expected or create dissonance. In the former case, it will reinforce the initial beliefs; in the latter, it will modify the beliefs about the system. Therefore, we propose that the result is the feedback from our decision. Feedback is the element that allows beliefs to evolve. The particularity of feedback is that it can take different forms to reflect the state of the element impacted by the system user's decision. The final result of the decision to trust or not can therefore be interpreted by external actors and formatted to reflect, from a certain point of view, the whole process put in place by the human or AI leading to the final decision.

The previous section highlighted most of the components of trust in AI. The aim was to understand the processes by which trust evolves, the indirect influences on its behavior, and how to measure trust with different tools that have their own characteristics. In the following section, we will focus on feedback and its central role in the evolution of trust, including ways of integrating a type of feedback to a specific temporality (during the task and afterward). We then propose to integrate these different types of feedback into a general model of trust evolution, based in part on models found in the literature. We end by proposing an applicative framework in the context of maritime surveillance, where we will see, in particular, their constraints, how they work, and how to integrate the previous theoretical parts into this specific case.

B. Feedback

In each of the models in the literature on trust in automation, the concept of outcome is present. This result is the element that enables the modification of a human's belief in his autonomous system. In particular, it is the result of human-induced behavior in the use or non-use of automation. In the Mayer & Davis model, for example, it is described as an outcome. In its sense, this term refers to performance feedback on whether the task has succeeded or failed. It is simplifying and does not allow us to describe the elements that can constitute feedback being more than a success or failure. Then we have Lee's & See model, where the outcome is described as a "display", i.e. it is an integral part of the system, since it returns the result of using the system via an interface. What we are proposing brings nuance to the possible feedback given to the user to modify his initial knowledge of the system. In this approach, feedback is strongly associated with user learning. It provides an assessment or creates dissonance in the learner concerning his use of a system and its integration into his environment. The definition of beliefs shows that they evolve according to the use of the system and its results. This includes the notion of learning, since beliefs are what humans perceive of the system, including its performance, process, and purpose.

Over time and interactions, this perception evolves, and so does his or her knowledge. As feedback is a central element of learning and beliefs are being directly linked to learning, it seems obvious for us to transpose the descriptions of feedback for learning in our case on trust. In Bosc Miné's [30] article, she describes the different types of feedback applied to various contexts and shows the advantages and disadvantages of each. For example, some feedback is more suitable for experts, while others are more suitable for novices. We find groups of factors such as Human, System, and Environment at the source of the different feedback. The operator can provide himself with auto-feedback, i.e. it necessitates a kind of expertise to provide himself with feedback to learn from something observed. The system can provide direct or delayed performance feedback on its use, e.g., you succeeded in doing this or failed in doing that. As far as the environment is concerned, several types of feedback can be provided like physical feedback, such as a car accident, or feedback provided by a teacher (not linked to the operator-system pair) during a learning phase. These types of feedback are not necessarily independent of each other and can all be provided for the same event. Let us take the example of an autonomous car accident, in which different temporalities may exist:

- Before the accident, the car's automated system can trigger indicators (alarms, for example) to warn of an incoming accident;
- Just after the accident, the operator will question himself or the system, via auto feedback, which has not reacted as he expected;
- Several days later, an expert having analyzed the car's black box will be able to tell the driver everything what was done wrong, and thus provide the driver with elaborated feedback.

To provide some guidance, we need to be able to describe all types of feedback and understand the role of each. These three main types are elaborated feedback, verification feedback, and elaborated verification feedback. An overview of the feedback tree is given in Fig.2.

- Elaborated feedback comes from an external source and is intentional, the aim is to train the receiver of the feedback by providing clues to guide the individual towards a correct response;
- Verification feedback is an information concerning the correctness or incorrectness of the response, it can be separated into intentional feedback, external unintentional feedback and auto feedback;
  - Intentional feedback comes from something or someone that is external to the individual, such as a teacher or a device enabling the transmission of information; it allows the learner to have a teacher directly providing the necessary information;
  - Unintentional external feedback is the direct consequence of natural interactions with the physical or social environment. They can provoke auto feedback in the individual, giving him or her food for self interpretation;
  - Auto feedback is generated by the individual directly during or at the end of the action. These are the individual's own feelings and thoughts about the processes involved in performing the task. Its impact is difficult to calibrate, given the subjectivity of thought.
- Elaborated verification feedback is a mixture of the two previous sub-categories. They provide the correct answer, information explaining why the given answer is incorrect and why the expected answer is correct. This makes it possible to compare the answer given with an expert answer. An example would be the case-based reasoning feedback. This takes the current given answer and searches the database for past (or expert) answers, proposing the nearest "neighbor" for comparison. It has the ability to recall what the expert had proposed in this case, with parameters as close as possible.
  Each context and task allow one to define the appropriate answer for each question. Customized solutions for each category of learners are also necessary to optimize learning.

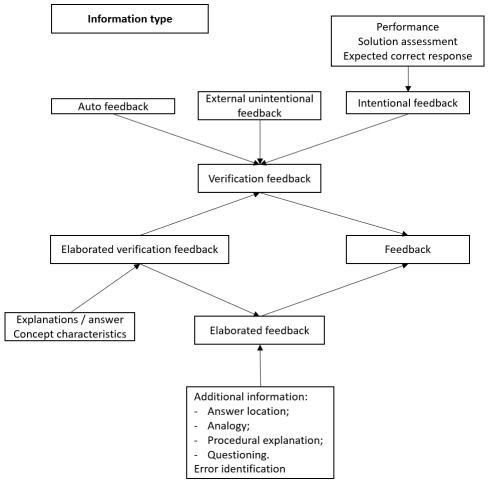


Fig. 2. Translation of the Bosc-Miné [30] model of feedback in learning. Feedback is divided into different sub-feedbacks. Each sub-feedback is associated with certain characteristics and contexts. The black arrow represents an heirloom. The source of the arrow is a subpart of the destination.

All these forms of feedback have common characteristics: from whom should it be given ? (itself,something,someone), how is it connoted ? (positively, negatively), what is focused on ? (task, process, self-regulation, person), presented with ? (alone, several), when should it be given ? (immediately, delayed), how should it be formalized ? (the solution, assessment criteria, individual characteristics).
Therefore, we propose to sort and integrate these different characteristics into a synthetic model of trust evolution. Our interest is to show that there are two temporalities in the evolution of trust and that different types of feedback need to be put in place to have a relevant effect on trust. This will be discussed in the next section.

III. Model

Our model incorporates elements from the literature such as the Parasuramann, Mayer, Lee & See and Hoff models, to which we add elements from the feedback description. Our interest lies in showing two types of feedback that affect trust over the course of interactions. These types of feedback act on two different temporalities, during the task and afterward (see Fig.3). The aim of immediate feedback is to maintain the

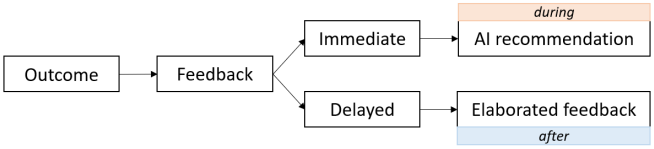


Fig. 3. Changes to the basic outcome. Feedback applies to two timeframes, each with its own specificities.

operator's awareness [18] on the mission. We want to avoid deterioration in mood or fatigue, which would degrade the operator's performance and make him less attentive to more critical tasks. Then, we have post-interaction feedback. This provides details on various elements encountered during the mission and highlights any disagreements. If the final state of an event is unknown, then this feedback would be constructed by an expert providing an inferred ground truth. This feedback incorporates the notion of learning.

There are a number of points to bear in mind. Immediate feedback must not be over-soliciting, i.e. it must not add workload on operator's in contexts that may already be overloaded. In terms of design integration, it must be simplistic and clear. It is also important to ensure that the first recommendations are not errors, as this could reduce the operator's trust to a level that would block further action [1]. For elaborate feedback, we need to be able to take into account the limitations of the field during experiments. Some knowledge are unknown, and we must not make the mistake of integrating it into experiments to establish whether a type of end-of-mission feedback is more effective than another. Being precise in the feedback certainly enables the learner to better integrate knowledge, but being as close as possible to the field gives a better understanding of the processes encountered.

We propose that immediate feedback should be in the form of a AI recommendation. Having direct feedback from the trustee seems to us to be important for the evolution of trust. This recommendation comes after the operator's initial choice and is there either to validate or refute the operator's initial decision. It fulfills the role of a decision-support AI, while the human still has the final decision. Therefore, this recommendation can influence human beliefs in AI. If the recommendation appears to be a mistake for the operator, then the beliefs through performance, process, and perceived goal will be degraded, while if the AI gives a recommendation in line with the initial decision, then its beliefs will be reinforced. No ground-truth information can be established at this stage of the interaction.

Regarding elaborated feedback, this takes as input all interactions captured by the system, including the event to be categorized (including all traces available up to that point), the operator's initial decision, the AI's recommendation, and the operator's final decision. Behavioral values such as reliance, compliance, and verification time are also captured. The aim is to compare, on the basis of all these factors, what decision an expert would have made in this specific case. This feedback contains a section on explicability that is essential for optimizing the way we explain what happened at a precise moment. The approach to explicability needs to be calibrated according to the target audience: some will prefer a mathematical approach, others a schematic or literary one.

Let's now turn to the integration of these terms into a model of trust evolution. The first phase occurs during interaction. A task is to be carried out, embodied either in a physical manifestation or via a computerized interface. In both cases, this task contains information that the AI can process so that it can make a recommendation. This recommendation, whether in dissonance or agreement with the initial decision, will modify the operator's beliefs (degrade or reinforce). The task will be added to a pre-existing or null task list. This addition increases the situational risk [16], as it increases the number of iterations to be processed. If the operator perceives this new iteration, then the perceived risk will also increase. There may be a dissonance between ground-truth and perceived risk. Perceived risk [2] is the only one that has an impact on a decision to trust or not. Next, we have the behavior related to trust. This behavior is affected by factors linked to the operator (such as perceived risk [16], self-confidence [31], workload [18], trust, beliefs), the environment (such as the event to be treated) and the system (such as the AI recommendation [10]). The elaborated feedback is fed by the trust-related behavior, the AI's recommendation, the initial decision, and the task information to be performed. As long as the mission is in progress, we have this interaction cycle.

The second phase takes place after the interaction. The elaborated feedback, previously fed by all the events that took place during the interaction, provides information to analyze any inconsistencies or dissonances. This feedback must be provided by an element external to the Trustor-Trustee pair, like another human or a device deemed effective. This feedback will have an impact not only on the trustor's beliefs towards the trustee but also on his self-confidence. It is a way of taking a step back and reflecting on the events that took place during the interaction.

In the following section, we propose to apply this model to the application context we will be experimenting with in the future: maritime surveillance.

IV. Maritime Surveillance Use Case

Maritime surveillance is a cognitively challenging work domain that involves high operational tempo and information overload. The crews of the maritime patrol aircraft are tasked with carrying out missions to control human trafficking, drugs, dangerous substances, illegal fishing, and sea rescue.

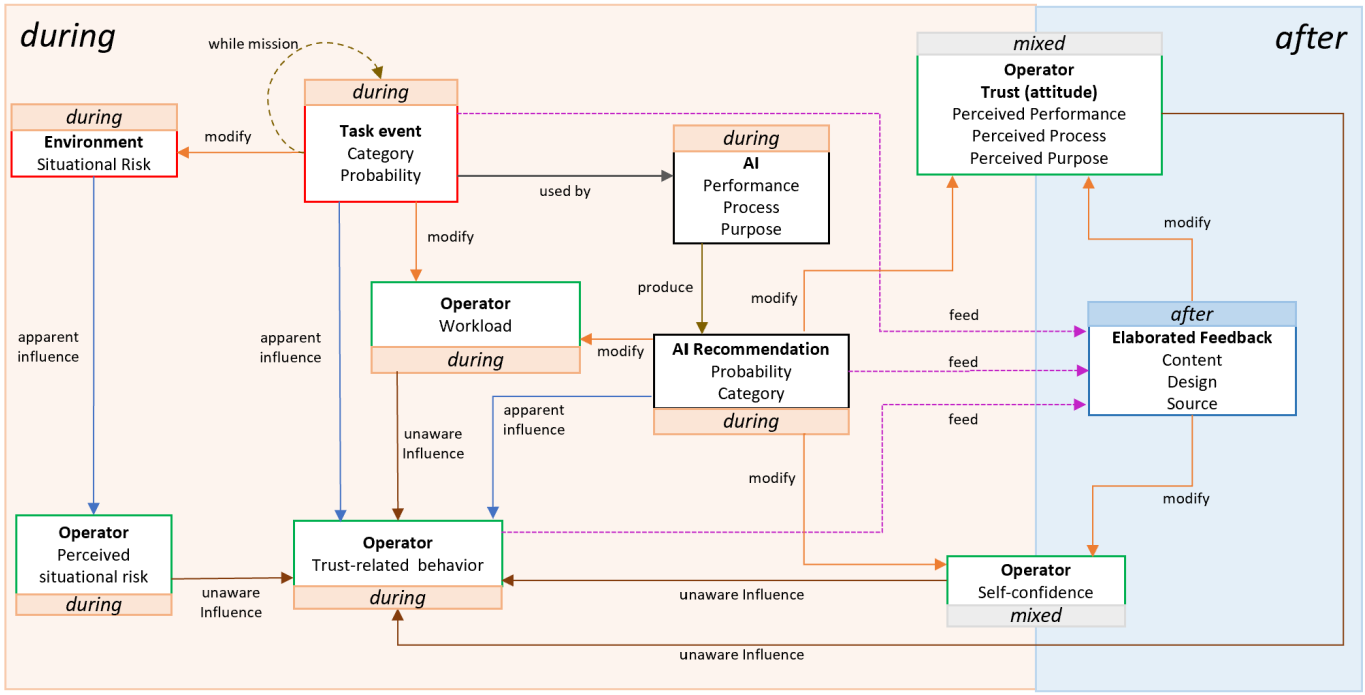


Fig. 4. Synthetic model of the evolution of trust. Two temporalities separate this model during (light orange) and after the task (light blue). The dynamics of this model arise from a task-related event. Red rectangles represent an environment-related state, green a human-related state, and blue a mixture of the three. Two types of arrow are visible. Solid arrows represent an influence on a process linked to the human, the system, or the environment. Orange is a modification of a state, blue is an apparent influence, i.e. perceived by the operator, and brown is a non-perceptible (or at least non-intellectualized) influence. The dotted arrows represent a transfer of information with no direct influence on any of the three actors.

Currently equipped with electronic surveillance equipment used manually by maritime surveillance operators, Thales is looking to integrating artificial intelligence tools to increase mission performance over the long term, to reduce the risk of missing illegal activities, and to increase operator resilience during high-pressure phases. To validate the value of such cooperation, we will re-use the application developed by Thales Defence Mission Systems (Thales DMS) and Thales Research Technology Canada (TRT Canada), which integrates Cognitive Shadow (TRT Canada) with AMASCOS (Thales DMS). AMASCOS is a tactical display that represents a situational context for maritime surveillance (see Fig.5). Cognitive Shadow (CS) [32] is a decision support AI composed of seven different supervised learning algorithms including Naive Bayes, Decision Tree, K-Nearest Neighbors, Support Vector Machine, Logistic Regression, Neural Network, and Random Forest. A configurable model evaluation and aggregation method is integrated to select the best recommendation amongst these algorithms. All these algorithms can be trained on classifications made by experts and developed in consultation with them to determine which information is relevant to a specific task (target classification in our case). In figure 5, the operator is represented by the aircraft in the middle of the map. A radar detection field is used to detect vessels characterized by fourteen features that are a mix of categorical and numerical attributes, which include: platform type, speed, speed change, stationary, length, friend list, AIS(on-off), nearest track distance, cluster size, coastal proximity, interception, sea lane deviation, heading change, and nationality. With this information, the operator must be able to make decisions about the category of vessels such as 'Allied', 'Neutral', or 'Suspect'. At the end of an operation, the operator team carries out debriefs to see if they have made any mistakes or if they have made any clarifications that could help them improve for the next time. The problem is that they have to wait until the end of the mission to realize whether or not they may have made mistakes. Here comes CS. It is integrated in such a way that it checks an operator's classification after an initial decision has been made. If the check does not agree with the operator's decision, and a certain probability threshold is exceeded, then CS provides a recommendation for the operator to re-check this target. This enables the operator to keep an eye on certain situations where he may be less attentive due to factors linked to himself (fatigue, stress) or to the environment (shake, noise, etc.). The recommendation creates dissonance in the operator, enabling him to refocus on the mission, to question himself, and to evolve his trust in the AI. In this very specific case, the elaborated feedback produced will be equivalent to the discussions operators have at the end of a mission to discuss disagreements concerning targets encountered. The point is to have a report to back up discussions and to show when the operator was right to change his mind or not. The ground-truth must be fed by another expert. The linguistic content will be generated procedurally

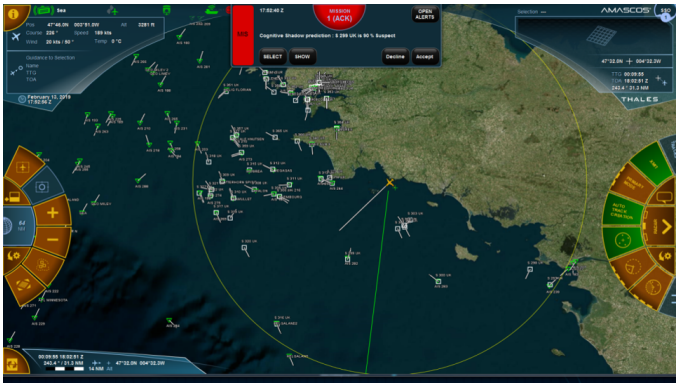


Fig. 5. AMASCOS interface representing an operational situation with a Cognitive Shadow recommendation showing that it is a suspect target.

to explain the data recovered during the mission.

V. Conclusion

The purpose of this study was to understand what trust is and what implications this has for the Human-AI pair when performing tasks in risky environments. The second aim was to introduce feedback terms into a model of trust evolution and to explain how different types of feedback can be introduced to optimize trust over time. For trust, we have drawn on general reference models covering all influencing factors, with Lee & See's [3] article as the main reference. Regarding feedback, the central article is by Bosc-Miné [30], in which feedback for learning is deconstructed to establish what can best be transposed to trust. We then presented a synthetic model of the evolution of trust, describing the main antecedents that have an impact on trust, or its behavior, over time. In particular, the perceived risk [16] described in Lee & See's [3] definition as a prerequisite for the study of trust. Concerning behavior, it is therefore associated with trust, but also with other factors such as self-confidence [31], workload [18], perceived risk [16], or the type of event to be dealt with. Two temporalities, during and after, allow us to target the types of feedback to be provided that will modify trust and self-confidence. The feedback from AI during the interaction will help the operator stay alert over time and reduce mental workload. Elaborated feedback from either the system (non-AI) or a person (external to the pair) helps to modify the operator's perceived beliefs (the basis of trust) of the AI and his own self-confidence. Other concepts related to trust have not been addressed, or only very briefly, such as explainability, transparency, acceptability, and so on. Explainability [7] is a major topic on its own with the goal of helping humans understand the basis of AI decisions. Different forms are being explored to determine which one will enable the most informed decisions to be made in the one-way relationship that is the Human-AI couple. Transparency, part of the explicability, aims to show what lies behind AI models. A definition of transparency provided by Chen [33] is "the descriptive quality of an interface pertaining to its abilities to afford an operator's comprehension about an intelligent agent's intent, performance, future plans, and reasoning process.". It has been shown that transparency can increase operator trust but, in return, can increase mental workload, which implies a design choice to best balance the impact of integrating transparency. Acceptability refers to the degree of acceptance of a tool's integration in a particular environment. It can be considered as an attitude or as behaviors [11]. The result is a level of trust in a tool. Beggiato et al. [11] show that acceptance evolves in the same way as trust during the learning phases. However, they are two different terms.

To the end, a case study applied to maritime surveillance was used to transpose the theory onto something more concrete. In this case, we will conduct experiments to study the evolution of trust over several weeks and validate this theoretical model proposed in this article. We will compare the impact of different types of feedback (immediate and delayed) and AI performance on operator trust over time.

References

[1] E. J. De Visser, P. J. Beatty, J. R. Estepp, S. Kohn, A. Abubshait, J. R. Fedota, and C. G. McDonald, "Learning from the slips of others: Neural correlates of trust in automated agents," *Frontiers in human neuroscience*, vol. 12, p. 309, 2018.

[2] R. Parasuraman and V. Riley, "Humans and automation: Use, misuse, disuse, abuse," *Human factors*, vol. 39, no. 2, pp. 230–253, 1997.

[3] J. D. Lee and K. A. See, "Trust in automation: Designing for appropriate reliance," *Human factors*, vol. 46, no. 1, pp. 50–80, 2004.

[4] M. Hou, G. Ho, and D. Dunwoody, "Impacts: A trust model for human-autonomy teaming," *Human-Intelligent Systems Integration*, pp. 1–19, 2021.

[5] K. E. Schaefer, J. Y. Chen, J. L. Szalma, and P. A. Hancock, "A meta-analysis of factors influencing the development of trust in automation: Implications for understanding autonomy in future systems," *Human factors*, vol. 58, no. 3, pp. 377–400, 2016.

[6] A. D. Kaplan, T. T. Kessler, J. C. Brill, and P. Hancock, "Trust in artificial intelligence: Meta-analytic findings," *Human factors*, vol. 65, no. 2, pp. 337–359, 2023.

[7] T. Saßmannshausen, P. Burggräf, J. Wagner, M. Hassenzahl, T. Heupel, and F. Steinberg, "Trust in artificial intelligence within production management–an exploration of antecedents," *Ergonomics*, vol. 64, no. 10, pp. 1333–1350, 2021.

[8] K. A. Hoff and M. Bashir, "Trust in automation: Integrating empirical evidence on factors that influence trust," *Human factors*, vol. 57, no. 3, pp. 407–434, 2015.

[9] S. Sousa and G. Beltrão, "Factors influencing trust assessment in technology," in *Human-Computer Interaction–INTERACT 2021: 18th IFIP TC 13 International Conference, Bari, Italy, August 30–September 3, 2021, Proceedings, Part V 18*, pp. 416–420, Springer, 2021.

[10] E. Avril, "Providing different levels of accuracy about the reliability of automation to a human operator: impact on human performance," *Ergonomics*, vol. 66, no. 2, pp. 217–226, 2023.

[11] M. Beggiato, M. Pereira, T. Petzoldt, and J. Krems, "Learning and development of trust, acceptance and the mental model of acc. a longitudinal on-road study," *Transportation research part F: traffic psychology and behaviour*, vol. 35, pp. 75–84, 2015.

[12] J. B. Lyons, G. G. Sadler, K. Koltai, H. Battiste, N. T. Ho, L. C. Hoffmann, D. Smith, W. Johnson, and R. Shively, "Shaping trust through transparent design: theoretical and experimental guidelines," in *Advances in Human Factors in Robots and Unmanned Systems: Proceedings of the AHFE 2016 International Conference on Human Factors in Robots and Unmanned Systems, July 27-31, 2016, Walt Disney World®, Florida, USA*, pp. 127–136, Springer, 2017.

[13] Y. Zhang, Q. V. Liao, and R. K. Bellamy, "Effect of confidence and explanation on accuracy and trust calibration in ai-assisted decision making," in *Proceedings of the 2020 conference on fairness, accountability, and transparency*, pp. 295–305, 2020.

[14] E. J. De Visser, M. M. Peeters, M. F. Jung, S. Kohn, T. H. Shaw, R. Pak, and M. A. Neerincx, "Towards a theory of longitudinal trust calibration in human–robot teams," *International journal of social robotics*, vol. 12, no. 2, pp. 459–478, 2020.

[15] S. Rice and D. Keller, "Automation reliance under time pressure.," *Cognitive Technology*, 2009.

[16] S. Hoesterey and L. Onnasch, "The effect of risk on trust attitude and trust behavior in interaction with information and decision automation," *Cognition, Technology & Work*, vol. 25, no. 1, pp. 15–29, 2023.

[17] T. Sato, Y. Yamani, M. Liechty, and E. T. Chancey, "Automation trust increases under high-workload multitasking scenarios involving risk," *Cognition, Technology & Work*, vol. 22, pp. 399–407, 2020.

[18] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "Situation awareness, mental workload, and trust in automation: Viable, empirically supported cognitive engineering constructs," *Journal of cognitive engineering and decision making*, vol. 2, no. 2, pp. 140–160, 2008.

[19] S. C. Kohn, E. J. de Visser, E. Wiese, Y.-C. Lee, and T. H. Shaw, "Measurement of trust in automation: A narrative review and reference guide," *Frontiers in psychology*, vol. 12, p. 604977, 2021.

[20] T. Rieger and D. Manzey, "Human performance consequences of automated decision aids: The impact of time pressure," *Human factors*, vol. 64, no. 4, pp. 617–634, 2022.

[21] J.-Y. Jian, A. M. Bisantz, and C. G. Drury, "Foundations for an empirically determined scale of trust in automated systems," *International journal of cognitive ergonomics*, vol. 4, no. 1, pp. 53–71, 2000.

[22] R. C. Mayer and J. H. Davis, "The effect of the performance appraisal system on trust for management: A field quasi-experiment.," *Journal of applied psychology*, vol. 84, no. 1, p. 123, 1999.

[23] K. Akash, W.-L. Hu, N. Jain, and T. Reid, "A classification model for sensing human trust in machines using eeg and gsr," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 8, no. 4, pp. 1–20, 2018.

[24] T. Xu, A. Dragomir, X. Liu, H. Yin, F. Wan, H. Wang, *et al.*, "An eeg study of human trust in autonomous vehicle basing on graphic theoretical analysis," *Frontiers in Neuroinformatics*, p. 70, 2022.

[25] Y. Lu and N. Sarter, "Eye tracking: a process-oriented method for inferring trust in automation as a function of priming and system reliability," *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 6, pp. 560–568, 2019.

[26] K. Akash, W.-L. Hu, T. Reid, and N. Jain, "Dynamic modeling of trust in human-machine interactions," in *2017 American Control Conference (ACC)*, pp. 1542–1548, IEEE, IEEE, 2017.

[27] S. W. Jaffry and J. Treur, "Comparing a cognitive and a neural model for relative trust dynamics," in *Neural Information Processing: 16th International Conference, ICONIP 2009, Bangkok, Thailand, December 1-5, 2009, Proceedings, Part I 16*, pp. 72–83, Springer, 2009.

[28] E. T. Chancey and M. Politowicz, "Designing and training for appropriate trust in increasingly autonomous advanced air mobility operations: A mental model approach: Version 1," 2020.

[29] O. Vereschak, G. Bailly, and B. Caramiaux, "How to evaluate trust in ai-assisted decision making? a survey of empirical methodologies," *Proceedings of the ACM on Human-Computer Interaction*, vol. 5, no. CSCW2, pp. 1–39, 2021.

[30] C. Bosc-Miné, "Caractéristiques et fonctions des feed-back dans les apprentissages," *L'Année psychologique*, vol. 114, pp. 315–353, June 2014.

[31] J. D. Lee and N. Moray, "Trust, self-confidence, and operators' adaptation to automation," *International journal of human-computer studies*, vol. 40, no. 1, pp. 153–184, 1994.

[32] D. Lafond, K. Labonté, A. Hunter, H. F. Neyedli, and S. Tremblay, "Judgment analysis for real-time decision support using the cognitive shadow policy-capturing system," in *Human Interaction and Emerging Technologies: Proceedings of the 1st International Conference on Human Interaction and Emerging Technologies (IHIET 2019), August 22-24, 2019, Nice, France*, pp. 78–83, Springer, 2020.

[33] J. Y. Chen, K. Procci, M. Boyce, J. L. Wright, A. Garcia, and M. Barnes, "Situation awareness-based agent transparency," tech. rep., ARMY RESEARCH LAB ABERDEEN PROVING GROUND MDUNIVERSITY OF CENTRAL FLORIDA ..., 2014.

Human-Machine Teaming For UAVs: An Experimentation Platform

Laila El Moujtahid
AI Redefined
Montreal, Canada
laila@ai-r.com

Sai Krishna Gottipati
AI Redefined
Montreal, Canada
sai@ai-r.com

Clodéric Mars
AI Redefined
Montreal, Canada
cloderic@ai-r.com

Matthew E. Taylor
AI Redefined
Montreal, Canada
matt@ai-r.com

***Abstract*—Full automation is often not achievable or desirable in critical systems with high-stakes decisions. Instead, human-AI teams can achieve better results. To research, develop, evaluate, and validate algorithms suited for such teaming, lightweight experimentation platforms that enable interactions between humans and multiple AI agents are necessary. However, there are limited examples of such platforms for defense environments. To address this gap, we present the Cogment human-machine teaming experimentation platform, which implements human-machine teaming (HMT) use cases that features heterogeneous multi-agent systems and can involve learning AI agents, static AI agents, and humans. It is built on the Cogment platform and has been used for academic research, including work presented at the ALA workshop at AAMAS this year [1]. With this platform, we hope to facilitate further research on human-machine teaming in critical systems and defense environments.**

***Index Terms*—Human-Machine Teaming, Reinforcement Learning, Multi-agent systems**

I. Introduction

Embodied AI agents, such as unmanned aerial vehicles (UAVs, or drones), have the potential to revolutionize various industries, including transportation, agriculture, and security. However, these agents evolve in the physical world and, as such, can have dangerous effects, especially when left unsupervised. For instance, a UAV may malfunction or fail to identify potential hazards, resulting in property damage or even human injury. Moreover, embodied AI agents can make decisions based on algorithms that may not consider ethical, moral, or legal implications. Therefore, it is essential for humans to have the ability to exercise meaningful control [2] and oversight on these agents to ensure their safe and responsible use. Human operators can monitor and intervene in cases of system malfunction, assess potential risks, and make ethical or legal decisions in complex situations that require their judgment.

In addition to oversight, humans can play a critical role in helping embodied AI agents achieve their tasks through collaboration. For example, in the case of UAVs, human operators in control centers can provide real-time guidance and support, ensuring that they perform the desired functions accurately and efficiently. Furthermore, humans can act as teammates in the field, working alongside embodied AI agents to achieve complex goals that require both human judgment and machine precision.

Moreover, it is important to recognize that human-machine teaming (HMT), the ability for humans and embodied AI agents to create a bidirectional collaboration, is a key aspect of safe and effective use of AI. The design, training, validation, and operation of such AI agents cannot be done in isolation; it is important to consider how they fit into a larger system that includes them. Humans, in particular, as operators or as teammates, should be considered an integral part of this system from the beginning.

Beyond this bidirectional collaboration, embodied AI systems often fail to consider "moral responsibility" and "socio-technical" factors in their operation [2]. The concept of meaningful human control (MHC) was introduced by Santoni de Sio and van den Hoven to enable humans to influence the behavior of embodied AI agents [3]. However, the original definition of MHC is inconsistent because humans may lack the expertise or the knowledge to fully control AI systems effectively. Cavalcante Siebert et al. [2] proposed four additional properties to improve the original definition of MHC: an "explicit moral operational design domain," "appropriate and mutually compatible representations," "control ability and authority," and "explicit linkage between AI and human actions." Therefore, it is crucial to design an orchestration platform to combine meaningful human control and human-in-the-loop to ensure that AI systems are trained and operated in a way that aligns with human values, societal norms and ethical behavior.

Cogment HMT provides a platform to design and experiment with human-machine teams, in particular involving UAVs. Built upon our Cogment [4] platform, it addresses the challenges of orchestrating collaboration between automated decision-making systems, including AI agents, humans, and their access to data and their effects on their environment. The Cogment HMT experimentation platform currently uses one simulated environment, and can be easily adapted for more realistic simulations and real-world deployments. We describe it and its properties in Section III.

Using the Cogment HMT experimentation platform, AI practitioners can develop agents capable of collaboratively working with humans and learn from their knowledge and expectations, and consider factors such as meaningful human control, trust, and managing cognitive load and enable effective bidirectional human-machine collaboration. We describe early results in Section IV.

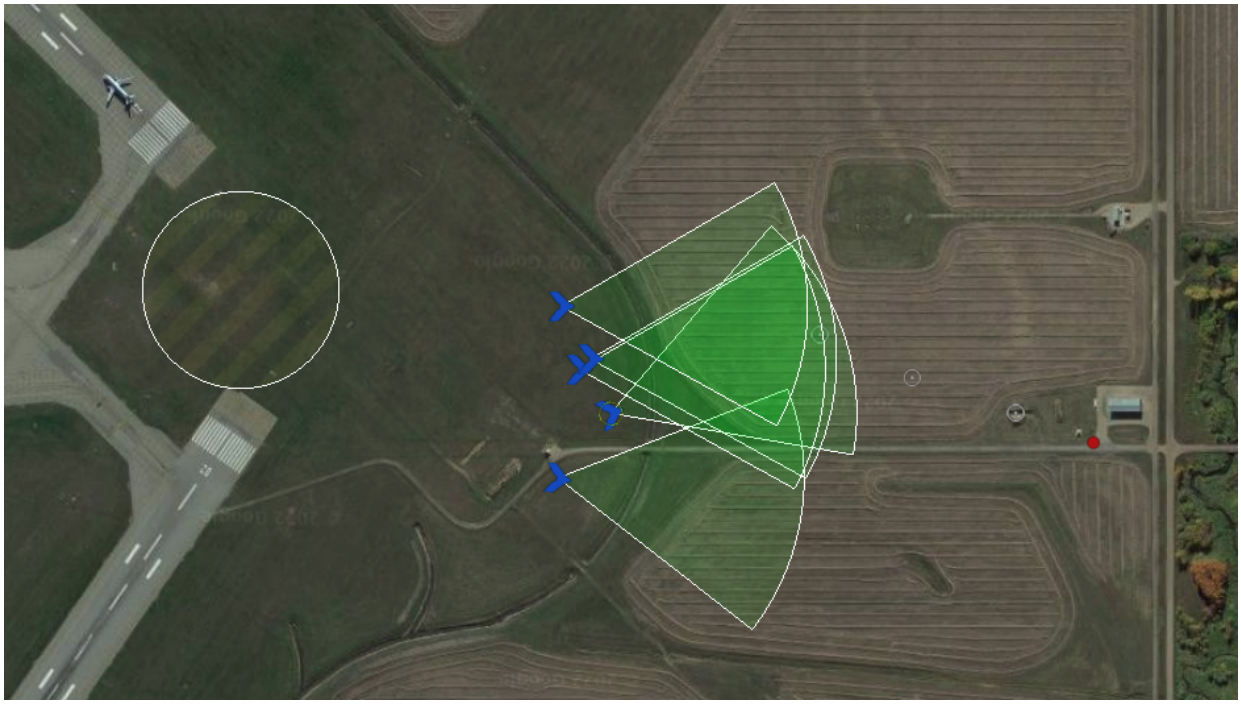


Fig. 1: This figure shows the main user interface for the Cogment HMT experimentation platform. The hatched circle on the left is a restricted area that the team of five *blue* agent defends. On the right side, the single red dot is the UAV attacker.

II. Related work

A. Autonomous robotics

Autonomous robotics has seen remarkable progress in recent years with the emergence of various popular experimental environments, including Gym [5], MuJoCo [6], and IsaacGym [7]. These environments are essential tools for researchers and developers to design and test algorithms quickly and easily in complex robotic ecosystems. Gym [5] is an open-source toolkit, developed by Open AI, and widely used in academia and industries. It provides a standardized interface and tools for creating, benchmarking, evaluating reinforcement learning algorithms. on a wide range of tasks, such as playing Atari games or controlling robots. Other relevant robotics simulation environments, such as Robosuite [8] built on top of MuJoCo, and IsaacGym, were developed to cover the need for physics-based simulation in robotics and computation requirements. To accurately model and simulate the physical interactions between robotic systems and their real-world environments, it is crucial for researchers and developers to model and capture the underlying physical laws and principles governing the behavior of objects in the real world such as gravity, friction, and collisions.

These experimentation platforms are complementary and well-suited for tasks that require a high degree of physical realism and significant computational resources, particularly for simulating large-scale robotic ecosystems. By using these benchmarks, researchers and developers can more easily identify and compare the strengths and weaknesses of different algorithms and select the most effective approaches in a controlled, safe, and cost-effective manner. However, these platforms focus on single robots, and are not suited for collaboration at scale.

B. Multi-agent Systems

Multi-agent reinforcement learning (MARL) [9] is a subfield of reinforcement learning (RL) [10] that focuses on learning algorithms for multi-agent systems. MARL is a relatively new area of research in artificial intelligence and robotics. In contrast to a single agent RL, MARL considers systems in which multiple agents interact with each other and the environment to achieve a common or individual goal [11]. Each agent operates independently and makes decisions based on its local observations, which may affect the observations and decisions of other agents [12]. Relatively few experimentation platforms support these types of systems, making it challenging for researchers and developers to test and evaluate their algorithms. PettingZoo [13] is an open-source experimentation platform. It supports heterogeneous and homogeneous multi-agent systems. It provides a collection of benchmark environments that simulate real-world scenarios, with a well-designed API and libraries, and agent interface tooling. However, when it comes to human teaming, these experimental environments have some limitations. Libraries, tools, and interfaces are not suitable for allowing agents to interact and collaborate with humans.

Beyond the platforms, multi-agent reinforcement learning (MARL) poses considerable challenges when it is time to design a robust and intelligent system [14]:

1) Scalability: The number of agents in a system can increase the complexity of learning and coordination between the agents, making it challenging to develop scalable algorithms for MARL.
2) Non-stationarity: The behavior of other agents in dynamic environments can change over time, making it difficult to develop learning algorithms that can adapt to these changes.
3) Communication and coordination: Designing adequate protocols and coordination mechanisms that can handle the heterogeneity of agents [15]. This becomes complicated when introducing robot-human social interaction and hardware requirements. The algorithms approaches need to be designed to handle the heterogeneous nature of the agents, as well as the dynamic and uncertain nature of the environments they operate in.
4) Partial Observation: An agent may not have access to complete information about the environment. The trade-off balance between exploration and exploitation can be delicate for each agent strategy while also exploiting their knowledge to achieve isolated tasks individually in collaborative interaction.
5) Reward function: Designing a reward function that encourages agents to work together while considering their heterogeneity and capabilities can also be a challenging task.

Finally, existing MARL platforms are designed to support relatively simple and small-scale environments, where agents have to learn simple tasks in homogeneous systems [15]. The success of MARL algorithms in one environment does not guarantee its success in another environment, making it challenging to develop algorithms or platforms that can bidirectionally transfer information or knowledge to different environments. Therefore, when it comes to designing more complex or realistic heterogeneous multi-robot or multi-agent systems, a more advanced platform with powerful computation and algorithmic capabilities is needed.

C. Human-Machine Teaming

Human-machine teaming (HMT) has become increasingly important in many fields, including robotics, aviation, and defense. This approach refers to the bidirectional collaboration between humans and multi-agent systems through a cognitive interface. There are several different approaches to training agents for HMT including the following human-in-the-loop learning (HILL) techniques:

1) Training from human demonstration refers to a method of teaching a machine how to perform a task by having a human demonstrate it. The machine can then observe and learn from the human's actions, and use this information to perform the task autonomously [16].
2) Training from human feedback, or reinforcement learning from human feedback (RLHF), refers to a process where humans provide guidance and correction to machines in order to evaluate the behavior of the system and provide feedback to help the machine improve its performance [17].
3) Human intervention refers to the practice of a human operator taking control of a machine in order to modify its behavior, either by overriding its actions or by enriching the learning with new tasks [18].

A comprehensive and detailed exploration of how humans can assist Reinforcement Learning agents in their learning process can be found in [19].

These approaches are applied in various applications, such as the Hanabi card game [20]. This game is used as a research tool to study collaboration and communication between humans and machines, as it requires players to work together to achieve a common goal. Originally designed as a benchmark for training cooperative agents, Hanabi has also been used to measure zero-shot training capabilities in collaborating with new players [21]. Ongoing research, such as the work done in Cogment-Verse [22], involves implementing the entire pipeline of self-play, training with randomly selected pre-trained agents, and testing and improving zero-shot coordination capabilities by involving human players during training or testing phases. Cogment's ability to efficiently switch between different actors, run multiple parallel trials, and support different training paradigms simultaneously has proven helpful in this endeavor. Another relevant example of a human-in-the-loop implementation is collaborative robots, or cobots [23]. These robots are designed to work alongside human operators in a collaborative environment, sharing tasks and responsibilities to improve efficiency, productivity, and safety. For example, cobots can perform repetitive tasks that may be difficult or unsafe for humans to perform on their own [24]. In other cases, cobots can assist human operators by handing them tools or materials, or by holding parts in place while the human performs assembly or other tasks. To achieve these results, these robots are equipped with sensors and other technologies that allow them to perceive the environment and interact with humans in a natural and intuitive way.

D. Multi-agent UAV Platforms

Only a few frameworks aim to facilitate research and development of UAV teams. Aerostack [25] is an open source software platform based on ROS [26] that enables developers to control and test UAVs in a specific context of inspection and surveillance. It interfaces with a variety of APIs and tools, incorporating a library of computer vision algorithms and sensors. It provides a modular, and customized multi layout architecture including modules level such as behavior, embedded controls and functional module interfacing through Aerostack APIs and services. While Aerostack provides strong equipment for human-robot interaction, there are some limitations when it comes to incorporate human feedback through HILL approaches. The platform provides useful tools and services, but it is unable to incorporate human feedback or human demonstrations. We believe such mechanisms are critical when transitioning from simulation to real-world applications.

Gazebo [27] is another 3D robot simulation software used for modeling and simulating robots, sensors, and environments. It is also an open-source software that is widely used in robotics research, and industrial robots. Gazebo provides a comprehensive set of tools and libraries for simulating UAVs, including flight controllers, sensors, and communication interfaces, and can be integrated with ROS for additional functionality. While Gazebo is a useful tool for simulating physic-robotic environments, it has its limitations when it comes to model complex UAV behaviors and social interactions within a multi-environment, such as interaction with operators and others UAVs in the real world, with a high degree of accuracy.

III. Platform

Cogment [4] is a general solution designed to build, train, and operate AI agents in simulated or real environments shared with humans. We used Cogment to implement an experimentation platform enabling AI research & development in scenario involving the collaboration between humans and AI-driven UAVs in defense and security use cases. Furthermore, the Cogment HMT experimentation platform provides a smooth path towards deployment in the real world.

A. Simulation

The Cogment HMT experimentation platform is designed to experiment with scenarios involving a teams of UAVs collaborating with humans. In its current form, the platform is focused on defensive scenarios to safeguard critical infrastructures. UAVs are operated from a ground control station by a human operator who focuses on high-level tasks such as threat detection and interception. Low-level control of the flight dynamics is not within the scope of this work — we currently assume the speed of computers will outperform a human's direct operation of a UAV.

The platform supports scenarios where two teams are present: the *red* team aims to penetrate a restricted zone, while the *blue* team aims at protecting this zone by intercepting the intruders. Figure 2 presents a schematic representation for these types of scenario.

Each team consists of one or multiple UAVs. Simplified flight dynamics are implemented for fixed-wing, vertical take-off and landing. Because flight control is not the focus of the platform, the modeling is simplified and projected into two dimensions. Each drone can be equipped with multiple sensors defined by their characteristics including range, orientation, and probability to detect. Finally, the UAVs can have a payload that can vary in their level of dangerousness — in our scenario, the red team has no payload and the blue team have electromagnetic pulse devices.

Additional fixed (e.g., ground-based) sensors can be added to the scenario to help the *blue* team with threat detection. They can have the same properties than the UAV sensors.

Each entity, UAV or fixed sensor, can be set up to have a full awareness of the other entities, automatically share information with their teammates, or only rely on their sensors. This allows researchers to consider offensive and defensive

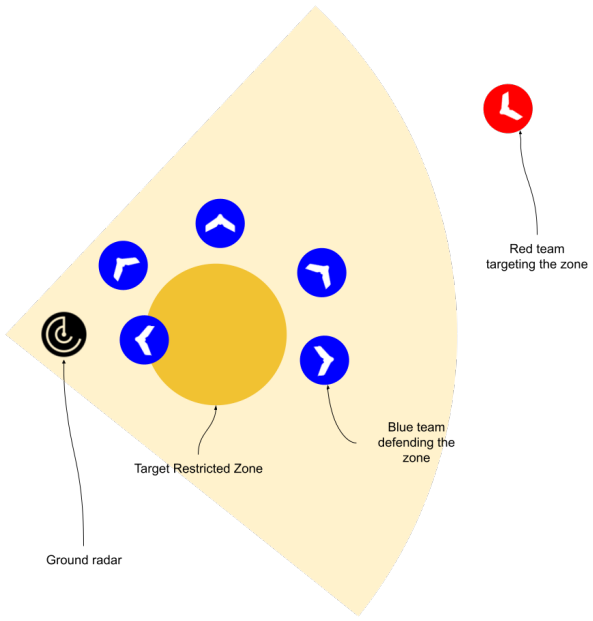


Fig. 2: This schematic representation of a typical scenario in the Cogment HMT experimentation platform showcases a *blue* team composed of 5 UAVs defending a restricted zone from intrusion by a *red* team of 1 UAV.

tasks as fully or partially observable, with or without communication. Entities' dynamics, sensory capabilities, and payloads are simulated in a dynamically configurable simulation that is integrated as an environment in a Cogment application.

B. Multi-agent architecture

The platform models the scenarios as a multi-agent system, described in Figure 3. The primary agent type controls the UAVs through velocity and rotation changes. Agents receive full or partial observations from the environment. Multiple implementations of UAV agents can be defined and instantiated, including path following, heuristic behaviors, or trained policies. Furthermore, the agent can dynamically switch between multiple control strategies over time.

The other type of actor enables human operators to have indirect control over individual UAVs. This allows a dynamic human-AI team-up in training or operation phases. The operator actor can add and remove waypoints to any UAV agent in their team. The inputs are then provided to the UAVs for decision making. For example, the operated UAV agent implementation follows a simple path following policy which consumes waypoints.

C. Platform use cases

The experimentation platform implementation choices are based on its use cases. To conduct the experiments described in Section IV, four use cases have been identified:

1) Record demonstrations from human operators who fully control the drones.

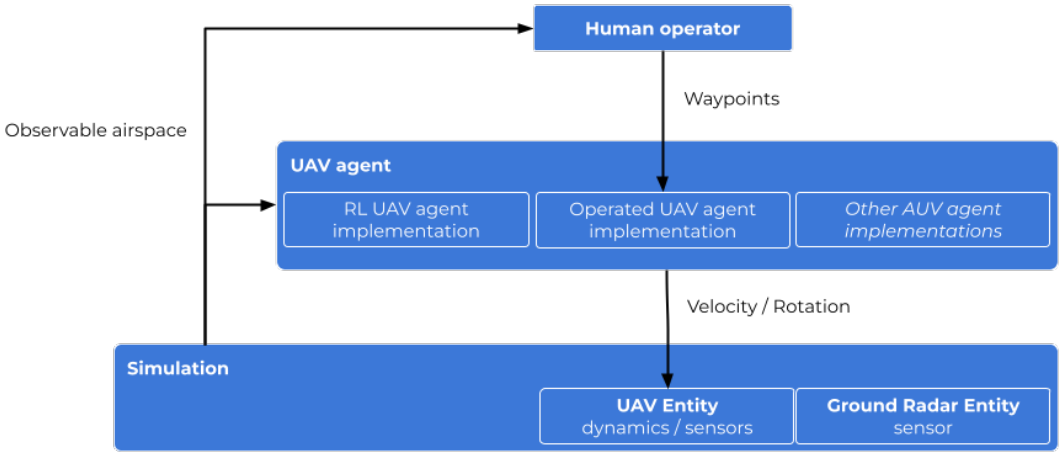


Fig. 3: Multi-agent architecture

2) Run MARL non-interactive batch training leveraging pre-recorded demonstrations.
3) Operate scenarios involving both human operators and trained AI agents for test and validation purposes.
4) Run MARL online involving human operators.

The experimentation platform runs episodes (i.e., instances of the airspace simulation), where one team is the winner at the end of every episode. Use Cases 1, 3, and 4 require running interactive episodes when human operators connect to the experimentation platform. Use case 2 requires the ability to run a large number of non-interactive (headless) episodes driven by the agents' training process. Use case 4 requires the ability to run batches of interactive episodes, and potentially headless episodes, and use the resulting data feed to continually train the agents.

D. Cogment-based implementation

From the identified use cases we designed and implemented the Cogment HMT experimentation platform architecture represented in Figure 4. Cogment is used to handle the orchestration of the execution and communication between the different components:

- The agents, encapsulated in dedicated µServices as Cogment actors
- The simulation, encapsulated in a dedicated µService as a Cogment environment
- The human operator UI, encapsulated as a client Cogment actor
- The training process, as a python script relying on the Cogment SDK to trigger trials and retrieve generated activity data
- The interactive session controller, as a component of the frontend, relying on the Cogment SDK to trigger episodes

Cogment dispatches observations of the environment from the simulation to the agents, as well as instructions from higher-level agents. It then dispatches agents' actions to the environment, which updates the simulation and agents' instructions. Furthermore, a priority-ordered list of multiple agents can be assigned to a single drone entity at once. If a higher priority agent outputs a velocity and rotation change, it overrides lower priority ones (e.g., enabling dynamic takeover by the human operator).

Cogment also provides its trial datastore, storing and making available the activity data generated by all actors in all environments.

The platform has been successfully deployed natively on workstations on both Linux, MacOS, and Windows. Longer headless training were executed on Compute Canada's slurm-based system. Finally, it was deployed on cloud infrastructure on AWS Canadian datacenter to experiment with human input.

E. Interactive Interface

The frontend part of the experimentation platform is a web application built in Javascript using React as its main framework. It consists of two different screens: an episode configuration form and an interactive episode runtime view.

The episode configuration screen enables users to configure different parameters. The episode runtime view in Figure 1 shows a map of the environment with a top view, the drones and their detection range, the ground radar, and detected red drones. Each drone can be selected by clicking on it; from there, users can add a waypoint for this drone by clicking on the map or remove existing waypoints. Finally, users can pan and zoom the map and pause or resume the simulation. This simple, yet polished user interface enables a high degree of interactivity and can be easily extended to support other tasks or collaboration modalities.

IV. Experiments

The experimentation platform was initially developed to support recent research [1]. In this work, we investigate the following two research questions:

1) How well does a trained agent perform in this specific environment?
2) Do agent or human demonstrations help to make the RL agent more sample efficient?

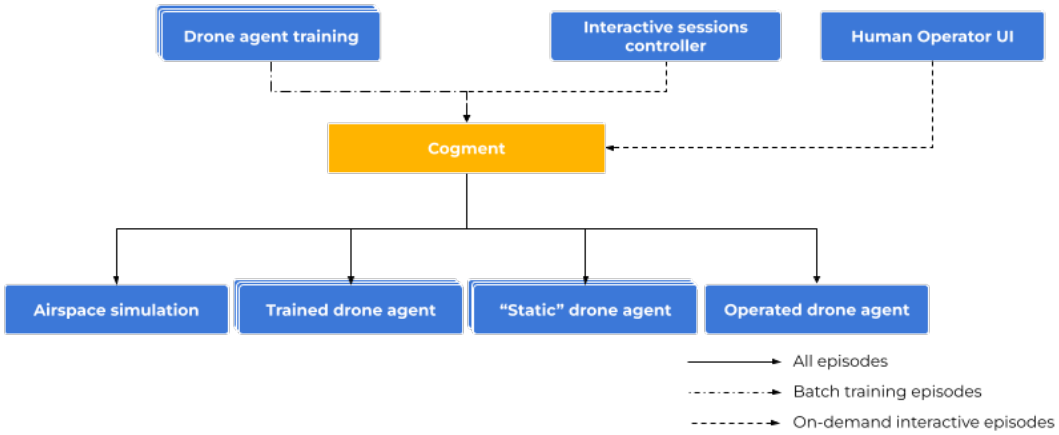


Fig. 4: Components architecture

In this section, we summarize this work to illustrate how the platform is leveraged.

We aim to train RL policies for the blue drone with the following Markov decision process formalization. The observation space consists of the relative positions of the drones and restricted airspace over three time steps. The action space is discrete and the reward function is positive for blue drones successfully neutralizing the red drone and negative if the red drone enters the restricted zone. The blue drones also receive a shaping reward proportional to their relative distance from the red drone in consecutive time steps.

The agents are trained using multi-agent centralized training and execution setup, with each agent having its own observation space and the same reward function and action space.

We evaluated the performance of the trained agent using the *success rate* as the metric. The success rate is defined as the percentage of times the blue team wins over all evaluation trials. This metric was chosen as it provides a clear and intuitive measure of the agents' ability to defeat the red drone and is directly proportional to the average reward. We run thirty evaluation episodes per hundred training episodes to compute the success rate. During the evaluation episodes, agents do not learn or explore. Our learning curves show the performance metric reported in the evaluation episodes averaged over five runs with different seed values to account for training and environmental stochasticity.

We train a D3QN agent (cite) and plot its performance in Figure 5. The agent reaches a success rate of roughly 90% in 3500 episodes. The trained agent outperforms the baseline heuristic agent, which has a success rate of 60%. The average human performance is around 63%, which is almost equal to the heuristic agent.

D3QN-PH reaches a success rate of more than 90% in 1600 episodes, outpacing D3QN, as shown in Figure 5. We performed an unpaired t-test to examine the significant difference between the performance of the D3QN-PH and D3QN agents. The D3QN-PH agents exhibit a statistically significant performance improvement compared to D3QN ($p < 0.0001$) and

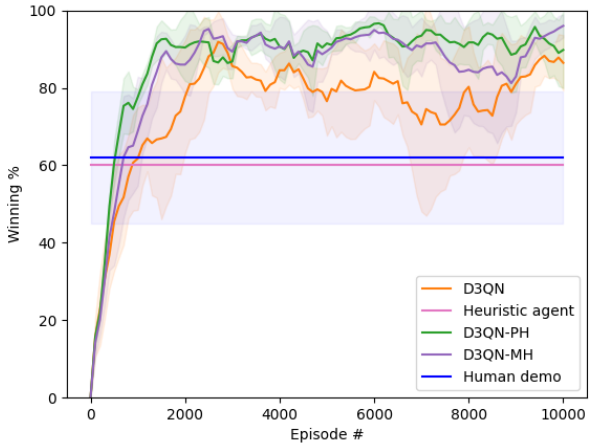
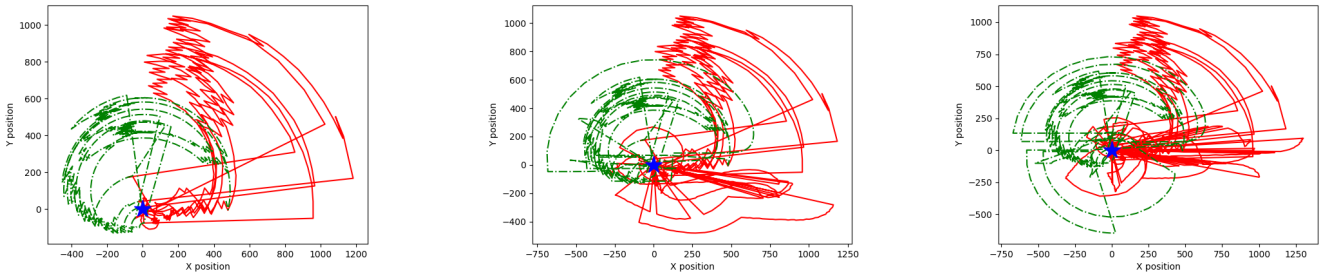


Fig. 5: The success rate comparison for D3QN, D3QN with trained agent demonstrations, D3QN with real human and trained agent mix demonstrations, and a heuristic baseline. Here, the suffix -PH represents demonstrations from a trained agent and -MH indicates a mixture of real and trained agent demonstrations.

the effect size is 1.43. At the end of learning, both algorithms converge to the same final performance level (around 90%). This supports our claim that trained agent demonstrations make the RL agent more sample efficient in our environment, consistent with existing results in the literature [28, 29].

A. *Pilot study using human demonstration*

We also trained the learning agent with a mix of agent and actual human demonstrations: D3QN-MH in Figure 5. We sampled an equal proportion of human and trained agent demonstrations in every mini-batch used for training. We used a mix of both types of demonstrations due to the lack of human demonstrations collected in our ongoing pilot study. The results do not suggest a significant learning improvement



(a) trained agent demonstration (b) a human user (User 1) (c) a human user (User 2)

Fig. 6: Visual representation of five episodes from trained agent and two different real human users

compared to D3QN-PH; however, the performance is still statistically significantly improved over the D3QN baseline.

We visualized five trajectories of all blue and red drones from trained agent demonstrations and two actual human demonstrations from different users who played more than 30 games, as shown in Figure 6. In this figure, the blue star denotes one of the ally drones that neutralized the enemy drone and is the frame of reference (located at $(0, 0)$). The red and green lines represent the relative position of the enemy drone and the restricted airspace (with respect to the blue drone's position). Figure 6(a) shows five trajectories of ally drones generated from the trained D3QN agent. We note that across all the figures, the red drone starts moving towards the restricted zone while being chased by the blue drones until it is neutralized. The low density of red lines around the blue star indicates that the blue drones quickly neutralizes the red drone without following it for a long time. From Figure 6(b) and 6(c), which depicts trials by two different human participants, we notice that there is more movement (high-density) around the blue star, suggesting that the human tries setting waypoints in different areas (using the whole team of five blue drones) of the map to neutralize the red drone. These trajectories are sub-optimal (longer trajectory length) as compared to the trajectories from trained agent demonstrations. However, these might be helpful to neutralize the red drones in difficult environment configurations where the trained RL agents fails to catch the enemy drone (trained agents have a failure rate of around 10% in this task).

V. Future Work

Since this simulation has a learning curve for humans because of the interface and the dynamics, we intend to collect more demonstrations from humans and to include a burn-in period for humans to understand the environment and learn to play before collecting demonstrations.

In order to advance research and development on human-machine teaming, we aim to continue expanding the experimentation platform.

We are working on accelerating the time to experimentation by providing access to off-the-shelf algorithms implementing well-known HILL learning techniques. This will allow researchers to focus on the core challenges of their research, rather than on implementing the basics of the learning loop.

We want to encompass a wider range of scenario types. This will enable researchers and developers to explore the potential of human-machine collaboration in more domains in defense and beyond, such as disaster response or logistics. One example of extension is the ability to support hybrid teams in the air including both UAVs and manned aircrafts.

Developing collaborative systems involving embodied AIs and human operators starts in simulation and aims at being deployed in the real world: the *sim-to-real* challenge. While the experimentation platform primarily aims at supporting *in silico* development and experimentation, by leveraging Cogment it is ready to move towards real world deployment.

Sim-to-real is not a one-step process from an agent developed in simulation to the real world. In fact, creating intermediate steps can make this process smoother and less challenging. Because it decouples the actor and environments' roles with their instances, Cogment makes this process smoother. The Cogment HTM experimentation platform leverages this feature to support both fully simulated episodes and interactive episodes, going from simulated *pseudo-human* actors to actual human actors. At the same time, it can go from its current simple airspace simulation to more realistic one and finally to pilot actual UAVs.

To ensure that the results obtained from our experimentation platform are reliable and can be compared across different studies, we aim at developing standardized evaluation methods. This is a challenging task in itself, as meaningful evaluation of human-machine teaming requires a holistic understanding of the role of humans in the loop. Evaluation metrics need to take into account not only task performance, but also aspects such as human trust, situation awareness, workload, and cognitive load.

Finally, we recognize that the interface between humans and AI agents is a crucial aspect of human-machine teaming. We are exploring how natural language interfaces can facilitate communication and collaboration between humans and AI agents, with the goal of improving trust, transparency, and robustness. This is a challenging research area, as natural language interfaces for complex systems require sophisticated natural language processing, dialogue management, and rea-

soning capabilities. However, we believe that this research can pave the way towards more effective and intuitive human-machine teaming interfaces in the future.

VI. Conclusion

The continued exploration and research around human-machine teaming systems has the potential to revolutionize a wide range of applications, from robotics to aerospace and defense. Emerging technologies such as large language models, MARL, and embodied robotics are playing an increasingly significant role in reinventing the **bidirectional collaboration** between humans and machines for decision-making in complex and critical environments [1].

In this context, the Cogment HMT experimentation platform provides a powerful and flexible tool for applied research and evaluation of HMT systems. Early results leveraged the main benefits of the platform including a simple and fast simulation, heterogeneous multi-agent architecture, the ability to run both headless and interactive episodes, and the dynamic agent "takeover" capability. The platform's flexibility enables researchers to easily test different configurations and make adjustments without affecting the overall functioning of the system.

We hope that this work and Cogment can facilitate the acceleration of human-machine teaming development while keeping the human factors at the center of this effort.

Acknowledgment

The authors would like to express their gratitude to the members of the Intelligent Robot Learning lab at the University of Alberta, JACOBB.AI, and Thales for their collaboration on this project. The authors would also like to acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada and the PROMPT program of the Quebec province.

References

[1] M. S. Islam, S. Das, S. K. Gottipati, W. Duguay, C. Mars, J. Arabneydi, A. Fagette, M. Guzdial, and M. E. Taylor, "WIP: Human-ai interactions in real-world complex environments using a comprehensive reinforcement learning framework," in *Adaptive Learning Agents Workshop, ALA 2023, Held as Part of the AAMAS 2023*, 2023.

[2] L. Cavalcante Siebert, M. L. Lupetti, E. Aizenberg, N. Beckers, A. Zgonnikov, H. Veluwenkamp, D. Abbink, E. Giaccardi, G.-J. Houben, C. M. Jonker, J. van den Hoven, D. Forster, and R. L. Lagendijk, "Meaningful human control: actionable properties for AI system development," *AI and Ethics*, May 2022.

[3] F. Santoni de Sio and J. van den Hoven, "Meaningful human control over autonomous systems: A philosophical account," *Frontiers in Robotics and AI*, vol. 5, Feb 2018.

[4] A. I. Redefined, S. K. Gottipati, S. Kurandwad, C. Mars, G. Szriftgiser, and F. Chabot, "Cogment: Open source framework for distributed multi-actor training, deployment & operations," *CoRR*, vol. abs/2106.11345, 2021. [Online]. Available: https://arxiv.org/abs/2106.11345

[5] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.

[6] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[7] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[8] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu, "robosuite: A modular simulation framework and benchmark for robot learning," 2022.

[9] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multiagent deep reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, vol. 33, no. 6, p. 750–797, Oct 2019.

[10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/the-book-2nd.html

[11] A. Wong, T. Bäck, A. V. Kononova, and A. Plaat, "Deep multiagent reinforcement learning: Challenges and directions," *Artificial Intelligence Review*, pp. 1–34, 2022.

[12] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," *Handbook of reinforcement learning and control*, pp. 321–384, 2021.

[13] J. Terry, B. Black, N. Grammel, M. Jayakumar, A. Hari, R. Sullivan, L. S. Santos, C. Dieffendahl, C. Horsch, R. Perez-Vicente *et al.*, "Pettingzoo: Gym for multi-agent reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 15 032–15 043, 2021.

[14] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: a survey," *Artificial Intelligence Review*, Apr 2021.

[15] M. Bettini, A. Shankar, and A. Prorok, "Heterogeneous multi-robot reinforcement learning," 2023.

[16] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, vol. 1, 1988.

[17] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems*, vol. 30, 2017.

[18] S. Chernova and M. Veloso, "Interactive policy learning through confidence-based autonomy," *J. Artif. Int. Res.*, vol. 34, no. 1, p. 1–25, jan 2009.

[19] A. Bignold, F. Cruz, M. E. Taylor, T. Brys, R. Dazeley, P. Vamplew, and C. Foale, "A conceptual

framework for externally-influenced agents: An assisted reinforcement learning review," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 4, p. 3621–3644, Apr 2023. [Online]. Available: https://arxiv.org/abs/2007.01544

[20] N. Bard, J. N. Foerster, S. Chandar, N. Burch, M. Lanctot, H. F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, I. Dunning, S. Mourad, H. Larochelle, M. G. Bellemare, and M. Bowling, "The hanabi challenge: A new frontier for ai research," *Artificial Intelligence*, vol. 280, p. 103216, Mar 2020.

[21] H. Nekoei, A. Badrinaaraayanan, A. Courville, and S. Chandar, "Continuous coordination as a realistic scenario for lifelong learning," *arXiv:2103.03216 [cs]*, Jun 2021. [Online]. Available: https://arxiv.org/abs/2103.03216

[22] S. K. Gottipati, L.-H. Nguyen, C. Mars, and M. E. Taylor, "Hiking up that hill with cogment-verse: Train & operate multi-agent systems learning from humans," in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, 2023.

[23] V. V. Unhelkar, S. Li, and J. A. Shah, "Decision-making for bidirectional communication in sequential human-robot collaborative tasks," *Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, Mar 2020.

[24] K. Kassem and F. Michahelles, "Exploring human-robot interaction by simulating robots," 2022. [Online]. Available: https://dl.gi.de/handle/20.500.12116/39622

[25] J. L. Sanchez-Lopez, R. A. S. Fernández, H. Bavle, C. Sampedro, M. Molina, J. Pestana, and P. Campoy, "Aerostack: An architecture and open-source software framework for aerial robotics," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2016, pp. 332–341.

[26] D. Thomas, W. Woodall, and E. Fernandez, "Next-generation ROS: Building on DDS," in *ROSCon Chicago 2014*. Mountain View, CA: Open Robotics, sep 2014. [Online]. Available: https://vimeo.com/106992622

[27] "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE, pp. 2149–2154.

[28] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband *et al.*, "Deep q-learning from demonstrations," in *AAAI*, 2018.

[29] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 6292–6299.

AI-Driven Expert Modeling for Cognitive Assistance in Anti-Submarine Warfare

Tanya S. Paul
Thales Research and Technology
Québec, Canada
tanya.paul@thalesgroup.com

Daniel Lafond
Thales Research and Technology
Québec, Canada
daniel.lafond@thalesgroup.com

Alexandre Marois
Thales Research and Technology
Québec, Canada
alexandre.marois@thalesgroup.com

Yannick Alcaraz
Thales Defence Mission Systems
Valbonne, France
yannick.alcaraz@thalesgroup.com

Sébastien Deries
Thales Defence Mission Systems
Valbonne, France
sebastien.deries@thalesgroup.com

Gabriel Jourdon
Thales Defence Mission Systems
Valbonne, France
gabriel.jourdon@thalesgroup.com

***Abstract*—Anti-Submarine Warfare (ASW) missions are becoming more complex with the increase of: i) collaboration between platforms, ii) multiple underwater platform types, iii) volume of data, vi) underwater sensor resolution, and v) challenges in detecting enemy submarines. Such complex and uncertain environments engender potential for cognitive overload and human error, leading to reduced situation awareness and detection performance. The objective of this work is to improve Human-Machine Interface through the use of a decision-support system for ASW crew, especially focusing on the sonar operator, sonar coordinator and underwater warfare officer roles. It aims at helping sonar operators taking the right decision at any decisive moment without disturbing the ASW drill. The use case investigated here focuses on modeling expert decision-making processes for determining active sonar transmitter/receiver immersion parameters. Policy capturing is a method which allows training models based on expert decision-making to infer and reproduce their judgment patterns. In this paper, we demonstrate how we incorporated a full-machine learning pipeline for ASW sonar parameter predictions based on cognitive modeling. Our study was able to generate a first proof-of-concept model based on supervised multi-output machine learning algorithms: Decision-tree, Linear-Regression, Random-Forest and K-Nearest Neighbour. Our work demonstrates the possibility of predicting sonar setting with accuracies up to and higher than 80% by using an online multi-model frugal learning approach requiring only a few hours of querying to efficiently capture expert decision patterns.**

Keywords—Multi-output machine learning, Policy-capturing, Frugal learning, Decision-support system, Anti-submarine warfare.

I. INTRODUCTION

Extracting expert knowledge for configuring decision support systems (i.e. expert systems) has been a long-standing challenge in cognitive engineering [1]. A key issue is that verbal descriptions of one's own decision process may be biased and incomplete, due in part to the difficulty of becoming fully self-aware of implicit, procedural, or non-verbal knowledge. One solution for that is to infer experts' judgment policies using supervised machine-learning techniques based on either historical or hypothetical cases. Once learned, an inferred decision policy can be used to bootstrap human decision making in real-time dynamic environments to avoid potential errors that may be due to mental overload, fatigue, stress, or distraction [2]. A cognitive system can be described as a goal-oriented agent that uses knowledge about its environment to execute a specific task, and adjust its actions based on the goal. Humans are considered cognitive systems; yet advances in computational neuroscience technology and automated systems have started to help humans support their cognitive abilities and result in the development of so-called artificial cognitive systems: "systems that perform tasks normally associated with human cognition", and joint (human-machine) cognitive systems [3].

Such a development of decision-support systems (DSS) or operator aids is becoming crucial in the underwater warfare domain, which might answer some of the challenges the sonar operator, sonar coordinator and underwater warfare officer roles deal with [4-5]. Indeed, research shows that these operators often face cognitive challenges pertaining to attention deficits and fatigue [6]. In addition to operational efficiency improvement, it is also particularly relevant regarding knowledge sharing between operators, experienced and novice, as crew turnover is a key concern for all worldwide navies. Moreover, training centers could take advantage of such AI-powered technologies for enabling digital tutoring techniques. Previous work has also been targeting the development of DSS for anti-submarine warfare (ASW) mission management [7], tactical coordinator officer [8] and supporting commander decision-making within ASW context, which are crucial for enhancing mission effectiveness and survival of naval vessels [6]. For example, the Office of Naval Research (ONR) and the Defense Advanced Research Project Agency (DARPA) have been deploying significant efforts towards decision aids in naval and air ASW [7]. However, relatively little attention has yet been placed on the integration of policy-capturing and expert-modeling methods for naval defense decision-support applications [9-11].

II. Policy Capturing for ASW

A. Expert Modelling

Cognitive Shadow is a policy capturing DSS developed by Thales Research and Technology Canada in collaboration with Thales Defense Mission Systems France and with several academic partners, which enables the automatic online learning of artificial intelligence (AI) models based on expert knowledge [12]. This approach differs from data-mining methods that require large amounts of data to discover hidden relationships (e.g. use of a neural network for mission planning for ASW) [13]. Policy capturing focuses on constrained-complexity models set to learn human expert decision/behavior/judgment patterns (without overfitting noise). The approach enables training models based on very small amounts of data, drawing from existing frugal-learning techniques [12]. Indeed, sparse data is often a major issue for AI within the defense industry, with very limited amounts of open or available data due to sensitivity and distribution restrictions. This tool enables to capture decision patterns from experts.

As for ASW use cases, it can provide to sonar operators recommendations about sonar settings, more specifically most appropriate immersion about the Transmitter (Tx) and Receiver (Rx) arrays. The model is trained on expert reasoning and previous decisions of realistic operational data specific to ASW. This tool could facilitate and enhance human-machine interaction within an ASW complex environment, increasing operational efficiency, simplifying and accelerating decision making and minimizing cognitive overload, bias and risk of error [14-15]. Moreover, one of the challenges with such a system is to help overcome the difficulty with diverging expert judgments: How can we best reconcile different answers from experts about sonar settings in a given context in order to create an effective group-of-experts model?

B. Anti-Submarine Warfare and Sonar Operations

Sound waves are a major medium of information transmission in ASW due to high conductivity and speed of transmission. Underwater signals are highly exploitable as they are composed of sound waves within which vibration helps propagation even better than aerial propagation and transmission [16]. Sonar is an acronym for sound navigation and ranging (SONAR): sonar systems are a key component of ASW vehicles as it enables the extraction of highly important information for tactical decision within naval and underwater warfare environments. This facilitates the localization or detection of obstacles and threats, but also their classification [17]. It can capture information such as the position and speed of a submarine, which are crucial for tactical operational and mission management of ASW crews. Processing such information can be very complex as the propagation of sound is a function of many variables such as weather variation, temperature, geographical zone, seasons, salinity, types of underwater landscape and bottom, movements of biological organisms, current and whirlpools, tides and other fluctuating artifacts which might add noise to the signal [18-19].

There are two types of sonars. Active sonars are composed of a receiver (Rx) sonar and a transmitter (Tx) sonar, whereas passive sonars only receive signals to be more discreet and detect/track signals from other sources [17, 19]. In this work, we are interested in the active sonar, for which a chain of pre-processing is applied to enhance the signal and minimize the noise, to then be able to make a decision about the detection [18-19] by:

- Identifying a threat hidden behind noisy signal;
- Being able to measure the parameters of the signal (timing of the propagation, direction); and
- Identifying the characteristics of the threat (recognize the type of vehicles, estimate threat parameters) [16].

C. Bathycelerimetric Profile

Our use case focuses on the bathycelerimetry profile types, a major feature of the complex underwater environment for sonar, which uses bathycelerimetric data. Bathycelerimetry can be defined as the measure of the celerity for each sound wave based on the distance, depth or pressure, temperature and salinity of the water, but also multiple additional parameters such as bottom type, water density, chemical and biological factors. Therefore, different types of bathycelerimetry profiles are possible based on the zone, temperature, salinity of water and more [19]. For example, temperature decreases as the depth increases. Based on this information we classify the types of bathycelerimetry. The bathycelerimetric profile is a reflection of the depth-dependent temperature and salinity profiles, this enables the calculation of the speed of the acoustic waves. It is also representative of the environmental conditions at the time when the sound emissions are generated (e.g. same place, same season [19-20]).

The acoustic signature can be described as the temporal representation of sound pressure. It integrates all the frequencies generated by a sound source and enables the source to be characterized [20]. This sound helps the sonar operator decide on the type of threat and also helps in the localization of an object or entity. The celerity can be described as propagation speed in meter per seconds (m/s) of a wave-like phenomenon such as an acoustic wave. The velocity of a sound depends on the properties of the medium in which it propagates such as the temperature, salinity and pressure [20]. Within this context, the sonar operator uses two types of outputs from the sonar to make their decision about the acoustic signature, which from their headphone they hear the sound waves emitted by the target entity and received by their sonar Rx. In addition, they are also presented with acoustic images which are the visual representation of the bathycelerimetric information onto a screen.

D. Current Study

The objective of this study is to demonstrate that DSS could be used to assist sonar operators in deciding on the optimal immersions to place Tx and Rx. Thus, the goal is to enhance operator cognitive performance in detecting threats, by

enabling such systems to have access to similar metrics and information used by the sonar operators. Additionally, this sets a first multi-output models proof-of-concept, which could be integrated in larger ASW technologies and systems such as Smart Assist (see Fig. 1). Smart Assist is a digital assistant aiming at providing a better underwater situational awareness to sonar operators. It is integrated to the operational platform, with no impact on the operational doctrine, and provides live sonar settings recommendations that are reliable, understandable and trustable. Currently in development in the Thales Variable Depth Sonar (VDS) CAPTAS product, it will facilitate automated reporting, allow access to past mission reports, and give relevant recommendations at any decisive moments of the mission across the ASW crew. Real-time recommendations use the current situation and bathycelerimetric parameters, but are also based on AI analytics from an environmental database and large VDS crew knowledge and past experiences.

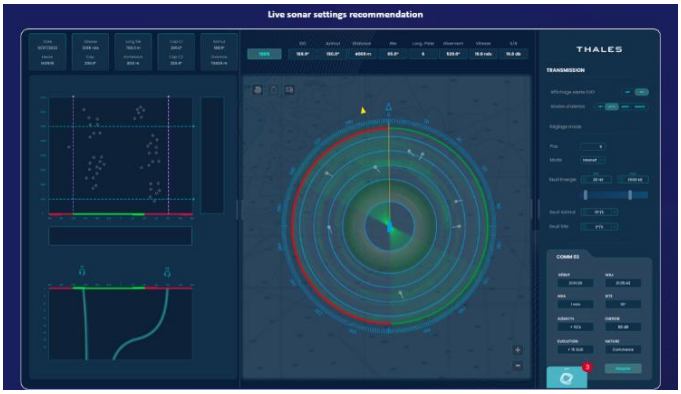


Fig. 1. Smart Assist interface integrated in the Thales variable depth sonar VDS human-machine interface.

III. Method

The work reported here involved two major steps: a) a workshop with domain experts such as Marine Officers and sonar operators in order to understand their challenges, pain points and desire for advancing operational use efficiency of sonar equipment; and b) an empirical policy-capuring data collection, with the development and deployment of our machine learning model for a first proof-of concept study.

A. Workshop

A workshop with subject-matter experts from the *Marine Nationale* was held, which included the following participants: commanding officers of underwater anti-submarine warfare operations of the *Marine Nationale* and *Commandant* (Commanding Officer) on the French vessels. We performed human factors interviews and discussions with experts to identify some of the main challenges, priorities and to express their needs for technological equipment and human-machine interaction within their operational context. More specifically, this workshop highlighted the need for support for sonar settings. Through this workshop, the marine experts highlighted some of the challenges including: dealing with a complex environment, fast filtering between false and true alarms, shift fatigue, complex sonar settings (+300), very long learning and skill improvements, getting and processing large amounts of data, and filtering the information.

This workshop enabled to better understand the environmental challenges and needs to tailor cognitive assistance, by understanding what types of decisions, and the factors which are impacted by the decision. This sets the first step in policy capturing and efficient development research in human-machine interaction for future operational deployment. Indeed, this workshop helped target the essential information needed and required for experts to make decisions, and in return limits the need for data mining and waste in computational process by only extracting relevant features from raw data. Additionally, it also helps reduce the number of examples or sample data to accurately model expert decision making.

B. Experiment

1) Participants: A total of five experts in ASW from Thales Defence Mission Systems with an average of 17 years of experience took part in the study.

2) Simulation and data collection: All five experts were presented with an interface, which simulates a real operational sound wave signal extracted from 20 different cases. Each case varied in sound wave characteristics and sound propagation. Cases varied according to the water depth, geographical zones, temperatures, and bathycelerimetric profile. For each case, participants were asked to write down the optimal immersion values for both Tx sonar and Rx sonar values based on two objectives: a) A detection zone considered as Far; or b) A smaller detection zone considered Medium. Due to the sensitivity of the settings, the ranges cannot be shared. Thus, the simulation would present them with both the imaging of the sound propagation and the acoustic signature.

Based on those two pieces of information, the sonar experts needed to make the decision on which immersion value pair (Tx-Rx) would maximize chances of detection of a submarine. Based on the subjective nature of this decision (there is no known objective measure of right and wrong, that is, no available ground truth), the experts were given the possibility to capture more than one answer (up to three) from the most to least optimal option. This is important, as there are up to 300+ possible settings and often experts do not end up with the same decision due to their different reasoning and decision-making pathways. The total amount of data collected can be presented as 20 cases × 5 experts, leading to 200 labelled cases with additional decisions in second and third top choices (for a total of 255 cases).

3) Machine learning pipeline: The machine learning method implemented relied on multi-output supervised models for regression (i.e., where the outputs are continuous variables) using the Scikit-Learn Python library. Multi-output models enable the DSS to predict decisions, which are composed of two interdependent parameters, which are Tx and Rx. The machine-learning pipeline is composed of several steps following the experimental design and the data collection, described in Fig. 2.

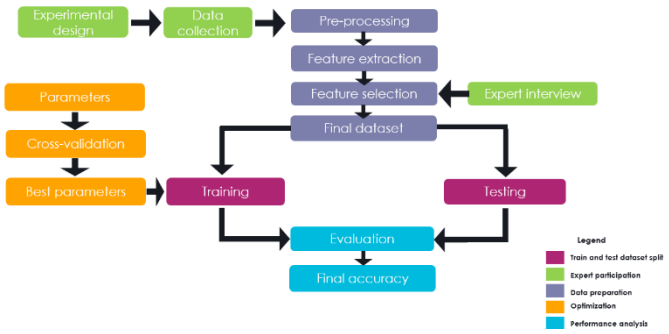


Fig. 2. Machine learning pipeline composed of data preparation (dark blue), modelling and testing (purple), optimization (orange), performance analysis (light blue), expert participation (green).

Once the data is acquired, our machine learning pipeline involves the following stages:

- *Pre-processing*: This stage consists in cleaning the data, and sonar signal as well as making sure our labels Tx and Rx are well distributed to avoid future bias;
- *Feature extraction:* Extracting higher-level information from the data such as bathy type or the most significant features from the dataset;
- *Feature engineering*: Transforming the data into easier processing format and normalize the data through methods such as one-hot encoding;
- *Feature selection*: Based on the most important factors considered by the experts, we select the appropriate features to be implemented into the model;
- *Modelling and training*: The four models used are Decision-tree, Linear-Regression, Random-Forest and K-Nearest Neighbor;
- *Optimization:* Once this is achieved, the dataset is split between training (with hyperparameter optimization) and testing, using a final 10-fold cross-validation to assess predictive accuracy on the held-out data.

a) Preprocessing and feature extraction: Preprocessing involves making sure the dataset is balanced and well distributed to avoid bias in Tx and Rx prediction. After collecting the data and running a preliminary analysis, some feature imbalances were detected (e.g. the geographical area did not sample the broad variability possible). For example, the majority of cases fell in one or two geographical areas. This might create some biases in the predictions of the model, and how the model trains and learns. Therefore, we applied a data augmentation strategy using plausible synthetic cases to diversify the database and reduce risks of bias. Data augmentation enabled to add cases with feature variants that were assumed not to impact decision outcomes. Secondly, this phase involved extracting the features, which are affecting both the acoustic signature as well as the sound waves imaging, perceived by the expert to make its decision. Features extraction can be represented in Table 1.

TABLE I. FEATURES EXTRACTION

Features	***Definition***	***Ranges***
Bathy Type	Represents the classification of the curve based on its celerity profile.	[0-5]
Temperature (Celsius)	The seven layers of depth-dependent temperatures, represented by 7 variables (e.g., Temperature1, Temperature2, …)	[25, -2]
Goal	Binary classification based on the distance ranges for detection: Medium or Far.	[0-1]
Velocity (ms)	The seven layers of depth dependent velocity (velocity at depth1, depth2, … depth7)	[1400-1550]
Geographical Zone	Geographical zone based on the type of sea or ocean.	[0-3]
Depth (m)	Represents the 7 layers of deep-water depth immersions.	[0-6000]

Note. Velocity values were rounded for sensitivity reasons.

b) Feature engineering and selection: Feature engineering was needed to transform some of the features extracted into the proper format using methods of encoding using sklearn.preprocessing LabelEncoder. Secondly, we only selected the most relevant features and dismissed the ones the experts did not take into consideration when making their decisions. This was conducted via a small interview (cf. Fig. 2).

Lastly, the selection process was enhanced using a deconflicting algorithm (Fig. 3). This algorithm was a necessary step to help in dealing with incompatible decisions across experts. This enabled the model to avoid having conflicting labels for the same case. We created a deconflicting function, which enabled us to converge toward a best choice based on two strategies (Fig. 4).

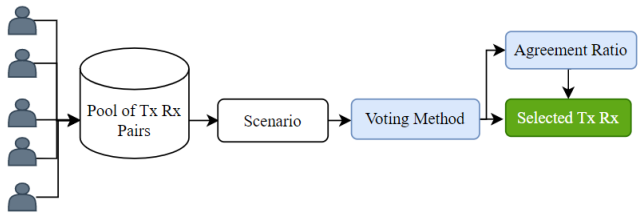


Fig. 3. Deconflicting pipeline.

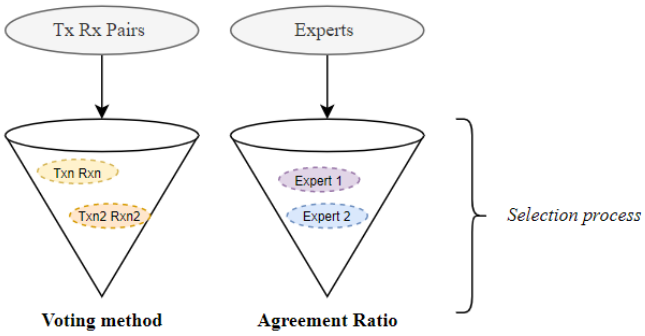


Fig. 4. Methods of de-conflicting: Voting method and Agreement ratio (from left to right).

The first one is by voting for the pair (Tx, Rx) with the most common agreement among all experts. The second strategy is based on an agreement ratio, which could be seen as a clustering on the average of responses of each expert. This means that if the algorithm has often selected an answer from one or two experts, which themselves means they agree with most other decisions, the algorithm will interpolate and decide on picking the answer of the expert the majority agreed upon. If no majority answer emerged, then the algorithm would select one based on the agreement ratio method.

c) Modeling and optimization process: The dataset is split between training and testing. The model training process includes hyperparameter selection through cross-validation following grid search, as well as cross-validation of the model accuracy. The model evaluation uses the 10-fold cross-validation process, sharing the dataset into 10 folds, at each iteration comprising 90% of the training data and 10% of test data. Lastly, the validation scores are used as the final model's performance, which are compared differently depending on the aggregation method selected. Therefore, the accuracy results from 10-fold cross validation and trained hyperparameters. Each set of hyperparameters from the machine-learning algorithms is also trained and cross-validated using a 10-fold rule. The best ensemble of hyperparameters used and last cross-validation are performed to determine the final accuracy. Lastly, to select the models, we used the "Best Aggregation" which defines the recommendation of Tx and Rx according to the model which is better at predicting all the different possible Tx-Rx pairs, using the average of their predictive accuracy. Examples of such methods have been also deployed and proved efficient using Cognitive Shadow [22]. This aggregation method helps select which of the four algorithms is then selected for recommendation.

C. Data Analysis

1) Expert homogeneity analysis: The expert homogeneity analysis enabled us to assess the degree of agreement over the decision across all five experts for each use case. In order to analyze the differences and variation in decision making across experts, we have developed an Expert Homogeneity Analysis metrics which is classified between three classes:

- *Low:* If all Tx-Rx pairs are different for all experts, and the selection process picked the one based on an agreement ratio.
- *Medium*: If there is a tie in voting methods between two Tx-Rx pairs, however the best agreement ratio expert is not within those two options.
- *High*: If most of the expert agreed on the same Tx-Rx pairs using the voting methods or if there is a majority for two Tx-Rx pairs and the best agreement ratio expert is within one of them.

Based on Table 2, we can observe a higher consensus over the goal with a Medium range than with Far. Global represents when the analysis was made regardless of the goal, which demonstrates that overall, there is still very high variation between experts in predicting and deciding the best Tx-Rx immersion parameters.

TABLE II. EXPERT HOMOGEINITY ANALYSIS

Goal	***Low***	***High***	***Homogeneity %***
Medium	6	14	70%
Far	8	12	60%
Global	14	26	65%

We observe overall a greater proportion of high cases of homogeneity, with a reduced homogeneity for the Far detection objective. However, we still note that the variation between both classes is only represented by a difference of 10%. The formula for the percentage of homogeneity can be calculated as shown in Equation (1) for each goal:

$$Homogeneity\ \% = \frac{\text{High} \times 100}{\text{High} + \text{Medium} + \text{Low}} \quad (1)$$

We observed no case where the expert was not part of the tied most-voted Tx-Rx pairs. This homogeneity metric is important as it helps demonstrate conflicting reasoning across experts. However, due to reasons of confidentiality, we are unable to share the statistical results of the variation of Tx-Rx pairs across experts on the different cases assessed.

2) Patterns and tendencies in features: Major patterns have been detected through the analysis of individual features. The first and major marker of the variability in feature is founded on the depth. As Fig. 5 demonstrates, the deeper the water, the higher variability and higher standard deviation are found across all uses cases. As for temperature, the deeper the water, the less variability is observed. Lastly, velocity does not demonstrate this correlation pattern. However, we note higher velocity within deeper water (i.e. for depth 7). This analysis enables determining how different components of the batycelemetric profile might affect model performance.

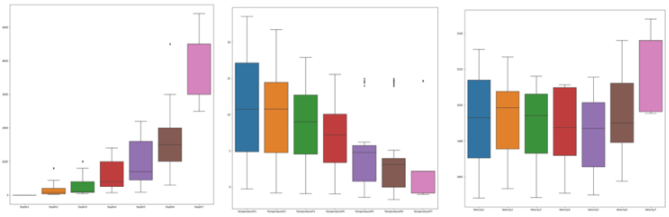


Fig. 5. Feature variability based on: i) Depth (1-7); ii) Temperature (1-7); and iii) Velocity (1-7), from left to right.

3) Models testing: In order to assess the performance of the different multi-output models, we considered two accuracy metrics on the validation data, R^2 (indicating the proportion of variance accounted for by the model) and the Root Mean Squared Error (RMSE), which represents the average absolute error (deviation) of the predictions.

IV. RESULTS

We compared the test accuracy between the four different models based on the goal (Far, Medium or Both combined). First, the results demonstrated a higher accuracy in predicting

Medium with 96.46% and closer range immersion metrics than Far with 85.89%. This is also aligned with the expert homogeneity analysis, where we detect higher consensus over closer ranges than for far ranges. For Medium ranges, the highest accuracy was achieved with the decision-tree model with an accuracy on validation data with 96.46% with an RMSE on validation of 4.74 meters. The second-best model was KNN with accuracy on test data of 90.63% with an RMSE on validation data of 11.79 meters.

For the Far detection goal, the highest performing model was a decision tree with 85.91% with an RMSE validity of 9.99 meters, closely followed by the KNN model with an R^2 of 85.89% and an RMSE of 13.80 meters. The second-best performance is attained by the random forest with an R^2 accuracy of 76.64% and an RMSE of 22.83 meters. Lastly, we can observe a poor performance for the Global models (combining Medium and Far cases), suggesting that attempting to simultaneously capture heterogeneous decision patterns in a single model can be counterproductive and may lead to poor generalization. Nonetheless, the KNN model succeeded in achieving a high global predictive accuracy of 99.94% with an RMSE of 1.22 meters. We observe a general higher predictability in Medium and Far over Global. There are higher discrepancies across models within Global compared to Medium and High, which would need further investigation. In addition, we would like to highlight that limited analyses were presented herein due to the sensitivity of the data.

TABLE III. MODEL PERFORMANCE

Goals	**Multi-Output Regression Models Accuracy**[a]			
	Linear Regression	***Decision -Tree***	***Random Forest***	***K-Nearest Neighbour***
Medium	85.37%	96.46%	83.16%	90.24%
Far	73.72%	85.91%	76.64%	85.89%
Global	29.99%	71.31%	43.57%	99.94%

[a.] Accuracy reflects the R^2 (in %) for the multi-ouput supervised machine learning regression algorithm.

V. DISCUSSION

The policy-capturing study reported herein aimed at demonstrating that experts' decision making for sonar parameter management can be accurately modelled using multi-output regression models. A group-of-experts model derived using this method can in turn be integrated as part of a real-time DSS system recommending Tx and Rx pairs to the user, yet keeping the human expert in the loop and in charge of decision making. Results demonstrate that despite the relatively small dataset, high accuracy (above a typical acceptability threshold of 80%) could be achieved using this policy-capturing method designed for learning with sparse data. While these results provide a compelling proof of concept for the feasibility of this approach, more work is needed in order to have a fully deployable mission-ready solutions.

Several future improvements are already under consideration. One first potential improvement would be to ensure a more diverse set of cases to ensure good sampling coverage of the parameter space, which is essential for good generalization in operations. A second potential improvement concerns our deconflicting methods which could be enhanced by taking into consideration years of experience, time in missions, or specifics about the experts' operational experience (e.g., mainly in one geographical area, or with one type of sonar). This first proof-of-concept study enables us to dive-in further within such methods and help target and optimize the next experimental design and data collection to produce a higher-maturity model. Additional parameters and features could be added in order for the machine-learning models to better understand the range of factors, as in other work such as [23].

Future work will also need to examine how to best provide explanations about recommendations to end users, in part to improve technology adoption but also to allow the human expert to review recommendations and decide to either accept or override them (which in turn can lead to further model improvement). Trust is a major feature for proposed digital AI-based techniques into operational doctrines.

Explainability features are also implemented within Cognitive Shadow, through the use of SHapley Addictive exPlanations or also called SHAP, which helps determine the weights of each feature or the coefficient of each feature for the prediction [24-25]. Yet, this component was not tested as part of the current study and, therefore, the experts could not receive any feedback from a DSS, nor any information with respect to the model's explanability. SHAP values could help improve transparency towards the types of recommendation and Tx-Rx predictions made by the system, a component that should be addressed in the future.

The intended practical application of this work is to insert such a learning and decision support module within the Smart Assist solution having a broader set of skills designed to bring a disruptive cognitive advantage to the ASW crew. The solution especially focuses on quickening the decision cycle of the sonar operator, sonar coordinator and underwater warfare officer roles and providing a security net to sustain nominal performance in adverse conditions. Expected operational benefits of the resulting decision-support capability are reduced mental load, improved situational awareness and enhanced decision speed/accuracy, leading to improved mission effectiveness and survivability for ASW, although DSS/automation integration must be carried out with caution [26-27].

Interestingly, the operational deployment of such a capability will also enable further data collection and allow iterative refinements and continuous adaption to the evolving operational environment. Given these possible outcomes, future work will also encompass the evaluation of mental workload, situation awareness and decision performance, compared with group-of-experts majority ground-truth responses, in order to test the operational benefits that such a DSS based on the principle of policy capturing could actually bring to the sonar experts.

ACKNOWLEDGMENT

Special thanks to Sandrine Heinis, Jean-Sébastien Thivierge and Jiye Li for their key contributions to this work. We are also grateful to Sébastien Tremblay and the Co-DOT lab at Université Laval for their support in testing and de-risking the Cognitive Shadow tool.

REFERENCES

[1] S.-H. Liao, "Expert system methodologies and applications—a decade review from 1995 to 2004," Expert. Syst. Appl., vol. 28, no. 1, pp. 93–103, Jan. 2005.

[2] J. S. Armstrong, "Judgmental Bootstrapping: Inferring Experts' Rules for Forecasting," in Principles of Forecasting. International Series in Operations Research & Management Science, vol. 30, J. S. Armstrong, Ed., Boston: Springer, 2001, pp. 171–192.

[3] D. D. Woods, "Cognitive Technologies: The Design of Joint Human-Machine Cognitive Systems," AI Mag., vol. 6, no. 4, Dec. 1985, Accessed: May 12, 2023. [Online]. Available: https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/511

[4] M. Kiziltan, "Cognitive Performance Degradation on Sonar Operator and Torpedo Data Control Unit Operator after One Night of Sleep Deprivation," Naval Postgraduate School, 1985. https://apps.dtic.mil/sti/citations/ADA161195 (accessed May 12, 2023)

[5] D. L. Robbins, "Decision-Making Process of an Antisubmarine Warfare Commander," Naval Postgraduate School, 1986. https://apps.dtic.mil/sti/citations/ADA176049 (accessed May 12, 2023).

[6] L. L. Merrill, L. J. Lewandowski, and D. A. Kobus, "Selective Attention Skills of Experienced Sonar Operators," Percept. Motor Skill, vol. 78, no. 3, pp. 803–812, Jun. 1994.

[7] W. Zachary, J. M. Ryder, and M. C. Zubritzky, "A Cognitive Model of Human-Computer Interaction in Naval Air ASW Mission Management," CHI Systems Inc., 1989. https://apps.dtic.mil/sti/citations/ADA217080 (accessed May 12, 2023).

[8] A. Leal, K. K. Chen, P. C. Gardiner, and A. Freedy, "Studies and Application of Adaptive Decision Aiding in Anti-Submarine Warfare," Perceptronics Inc., 1978. https://apps.dtic.mil/sti/citations/ADA057275 (accessed May 12, 2023).

[9] K. Labonté , D. Lafond, B. Chatelais, A. Hunter, F. Akpan, and S. Tremblay"Combining Process Tracing and Policy Capturing Techniques for Judgment Analysis in an Anti-Submarine Warfare Simulation," P. Hum. Fac. Erg. Soc., vol. 65, no. 1, pp. 1557–1561, Sep. 2021.

[10] L. Rothrock and A. Kirlik, "Inferring rule-based strategies in dynamic judgment tasks: toward a noncompensatory formulation of the lens model," IEEE T. Syst. Man Cyb., vol., 33, no. 1, pp. 58–72, Jan. 2003.

[11] K. Nokes and G. P. Hodgkinson, "Chapter 5: Policy-Capturing: An Ingenious Technique for Exploring the Cognitive Bases of Work-Related Decisions," in Methodological Challenges and Advances in Managerial and Organizational Cognition (New Horizons in Managerial and Organizational Cognition), vol. 2, R. J. Galavan, K. J. Sund, and G. P. Hodgkinson, Eds., Bingley: Emerald Publishing Ltd., 2017, pp. 95–121.

[12] D. Lafond, K. Labonté, A. Hunter, H. F. Neyedli, and S. Tremblay, "Judgment Analysis for Real-Time Decision Support Using the Cognitive Shadow Policy-Capturing System," in Human Interaction and Emerging Technologies. IHIET 2019. Advances in Intelligent Systems and Computing, vol. 1018, T. Ahram, R. Taiar, S. Colson, and A. Choplin, Eds., Cham: Springer, 2019, pp. 78–83.

[13] F. Wang, H. Tian, J. Jin, W. Yuan, and T. Zhang, "Research on Modeling and Application of Anti-Submarine Decision-Aided System Based on Neural Network," 2022 34th Chinese Control and Decision Conference (CCDC), Aug. 2022.

[14] K. Labonté, D. Lafond, A. Hunter, H. F. Neyedli, and S. Tremblay, "Comparing Two Decision Support Modes Using the Cognitive Shadow Online Policy-Capturing System," P. Hum. Fac. Ergon. Soc., vol. 64, no. 1, pp. 1125–1129, Dec. 2020.

[15] L. Grossetête, A. Marois, B. Chatelais, C. Gagné, and D. Lafond, "Active Learning for Capturing Human Decision Policies in a Data Frugal Context," in Machine Learning, Optimization, and Data Science. LOD 2021. Lecture Notes in Computer Science, vol. 13164, G. Nicosia et al., Eds., Cham: Springer, 2022, pp. 395–407.

[16] X. Lurton, "Acoustique sous-marine: présentation et applications," Editions Quae, 1998. Accessed: May 12, 2023. [Online]. Available: https://books.google.ca/books?hl=en&lr=&id=pZIHueqHyycC&oi=fnd&pg=PA9&dq=Acoustique+sous-marine:+pr

[17] M. Bouvet and M. Cresp, "Design Of Sonar Receiver And Oceanography," OCEANS 92 Proceedings@m_Mastering the Oceans Through Technology, pp. 455–459, 1992.

[18] M. Lamouret, "Traitements automatisés des données acoustiques issues de sondeurs multifaisceaux pour la cartographie des fonds marins," Université de Toulon, 2022.

[19] A. Mours, "Localisation de cible en sonar actif," Université Grenoble, Jan. 20, 2017. https://theses.hal.science/tel-01718704/ (accessed May 12, 2023).

[20] C. Persohn, L. Helloco, E. Baudiniere, and L. Martinez, "Recommendations to limit the impacts of manmade underwater acoustic emisions on marine wildlife," Ministère de la Transition Écologique et Solidaire, Jun. 2020.

[21] A. Marois, D. Lafond, A. Audouy, H. Boronat, and P. Mazoyer, "Policy Capturing to Support Pilot Decision-Making," Aviat. Psychol. Appl. Hum. Fac., vol. 13, pp. 26–38, Feb. 2023.

[22] A. Marois, K. Labonté, D. Lafond, H. F. Neyedli, and S. Tremblay, "Cognitive and Behavioral Impacts of Two Decision-Support Modes for Judgmental Bootstrapping," J. Cogn. Eng. Decis. Mak., Feb. 2023.

[23] L. Raillon and M. Fouquet, "UDT 2019 - Multistatic underwater protection sonar best patterns for harbour and larger critical environments." Accessed: May 12, 2023. [Online]. Available: https://www.udt-global.com/__media/libraries/platform-design/17---Louis-Raillon-and-Michel-Fouquet-Paper.pdf

[24] R. Galanti, B. Coma-Puig, M. de Leoni, J. Carmona, and N. Navarin, "Explainable Predictive Process Monitoring," 2020 2nd International Conference on Process Mining (ICPM), Oct. 2020.

[25] A. Holzinger, A. Saranti, C. Molnar, P. Biecek, and W. Samek, "Explainable AI Methods - A Brief Overview," in xxAI - Beyond Explainable AI, xxAI 2020. Lecture Notes in Computer Science, vol. 13200, A. Goebel et al., Eds., Cham: Springer, 2022, pp. 13–38.

[26] L. Bainbridge, "Ironies of automation," Automatica, vol. 19, no. 6, pp. 775–779, Sept. 1983.

[27] B. Strauch, "Ironies of automation: Still unresolved after all these years," IEEE Trans. Hum.-Mach. Syst., vol. 48, no. 5, pp. 419–433, Oct. 2018.

Une implémentation GPU de la méthode de recherche approximative FlyHash

Arthur da Cunha*
COATI, I3S & INRIA d'Université Côte d'Azur
Sophia Antipolis, France
arthur.carvalho-walraven-da-cunha@inria.fr

Emanuele Natale*
COATI, I3S & INRIA d'Université Côte d'Azur
Sophia Antipolis, France
emanuele.natale@inria.fr

Damien Rivet*
COATI, I3S & INRIA d'Université Côte d'Azur
Sophia Antipolis, France
damien.rivet@inria.fr

Aurora Rossi*
COATI, I3S & INRIA d'Université Côte d'Azur
Sophia Antipolis, France
aurora.rossi@inria.fr

***Abstract*—FlyHash is a locality-sensitive hashing algorithm inspired by the nervous system of the Drosophila fly. It has demonstrated to be particularly effective for similarity search, especially in the federated context where multiple players collaborate to solve a statistical learning task. FlyHash mainly relies on a process called winner-take-all, which is used to binarize information. However, the implementation of this process is a major challenge and limits the algorithm's usage in processing large data streams. In this paper, we propose a simple algorithm to make the winner-take-all operation efficient on GPUs. We create a FlyHash adaptation suitable for the CUDA architecture. We assess the speed of this version experimentally and present a comparison with the CPU version of FlyHash.**

***Index Terms*—Data mining, Hashing, winner-take-all, Distributed algorithms, Federated learning, GPU**

I. Introduction

Locality-sensitive hashing (LSH) is a technique in computer science introduced in [IM98] for hashing data so that similar data has a high probability of having similar hashes, while different data is likely to have different hashes. It is widely used, especially for similarity search in large databases using faster heuristics than traditional approaches, such as nearest neighbor searching. Locality-sensitive hashing is particularly effective in real-time applications, where the speed of similarity search is essential to handle a massive and continuous flow of incoming data.

In nature, animals are constantly faced with similarity recognition tasks. This is notably the case for fruit flies, which, when encountering new odors, seek to identify similarities with odors they have previously encountered in order to assess the potential quality of the available food. Observation of their olfactory nervous system revealed that some of these neural circuits bore striking similarities to well-known LSH algorithms. Based on these observations, researchers propose a new type of algorithm called FlyHash [SDN17].

FlyHash is based on the use of random projections followed by a binarization process, as is notably the case of one of the most well-known LSH heuristics SimHash [Cha02]. However, unlike SimHash, the binarization used by FlyHash is not based on thresholding, but on a process called winner-take-all (WTA), which we describe in detail below.

The adoption of winner-take-all is motivated on the one hand by its theoretical properties [YSRL11], and on the other hand for modelling the brain, in particular in the model of *Assemblies of Neurons* proposed by [PVM$^+$20] as well as in *Spiking Neural Networks* [Che17], artificial neural networks that are biologically closer to the real ones.

Besides, the classical winner-take-all implementation is known to represent a major bottleneck when one wants to process multiple data at once, or using large hashlengths (indeed the accuracy of such hashing algorithms improves significantly with the increase of the size of the generated hashes). In addition, the majority of data mining applications are now massively parallelized, via the use of distributed algorithms and the dominant hardware architecture is now the Graphics Processing Unit (GPU).

Our contribution with this work is to demonstrate that such WTA hashing schemes are compatible with the GPU architecture, allowing to implement them on a pipeline fully executable on GPU (Section III-B). Some works have focused on GPU implementation of winner-take-all including [MVSG$^+$09], but this direction remains relatively unexplored.

II. Motivations and applications

The FlyHash has been first studied in [SDN17], based on empirical observations of the functioning of the olfactory system of a fruit fly. In this section, we first provide a brief summary of such biological observations. We refer the reader to [SDN17] for a more detailed description. A formal description of the FlyHash algorithm inspired by these observations, together with our new implementation, is given in section

This work has been supported by the French government, through the UCA DS4H Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-17-EURE-0004, and through the AID INRIA-DGA agreement n°2019650072. The authors are grateful to the OPAL infrastructure from Université Côte d'Azur for providing resources and support.

* Authors are listed in alphabetical order.

III-A. Subsequently, we illustrate two important applications of FlyHash in Machine Learning, namely in an efficient and secure classification scheme in Federated Learning and as a potential subroutine to speed up the training of neural networks.

a) Neurobiological basis for Flyhash: The neurons which are activated by a given odor are determined by a three step procedure which is exemplified in Fig. 1. The first step

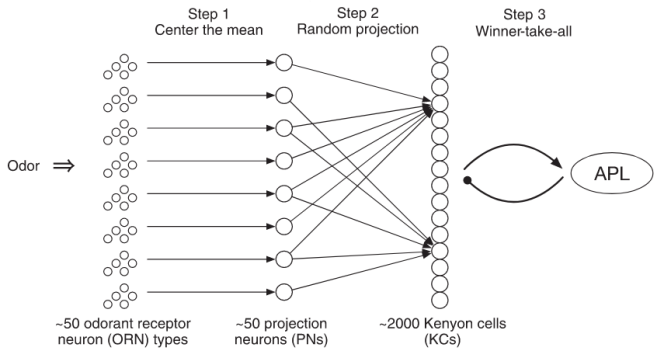


Fig. 1. The three steps of the olfactory system of a fruit fly as illustrated in [SDN17].

consists in direct nervous connections from odorant receptor neurons (ORNs) in the fly's nose to projection neurons (PNs) in particular structures called glomeruli. There are circa 50 ORN types which are activated by different odors. Thus, each input odor can be thought as having a corresponding location in a 50-dimensional space determined by the 50 ORN. For each odor, the distribution of ORN firing rates across the 50 ORN types follows an exponential distribution with a mean that depends on the concentration of the odor [SDN17]. For the PNs, such distribution of firing rates across the 50 PN types is also exponential, but for all odors and all odor concentrations the mean is approximately the same. The second step, which is where neurobiology offer us the main algorithmic insight, then involves an expansion in the number of neurons by a factor roughly 40: Fifty PNs project to 2000 so-called Kenyon cells (KCs), connected by a sparse binary random connection matrix, with each KC receiving and summing the firing rates from about six randomly selected PNs. The final third step is then performed by strong inhibitory connections from a single inhibitory neuron, called APL (anterior paired lateral neuron), which results in the aforementioned winner-take-all (WTA) operation. As a consequence, all but the highest-firing KCs are silenced, with the firing rates of the still-active KCs corresponding to the neurons activated by the input odor.

b) kNN-like classification in Federated Learning: FlyHash has recently found an important application as the basis of the FlyNN algorithm introduced in [SR21], who exploited the work of [DSSN18], taking advantage once again of biological observations, to design a classification algorithm (see Fig. 2).

FlyNN has been deployed in the context of federated learning in [RS22] and is currently the state of the art ap-

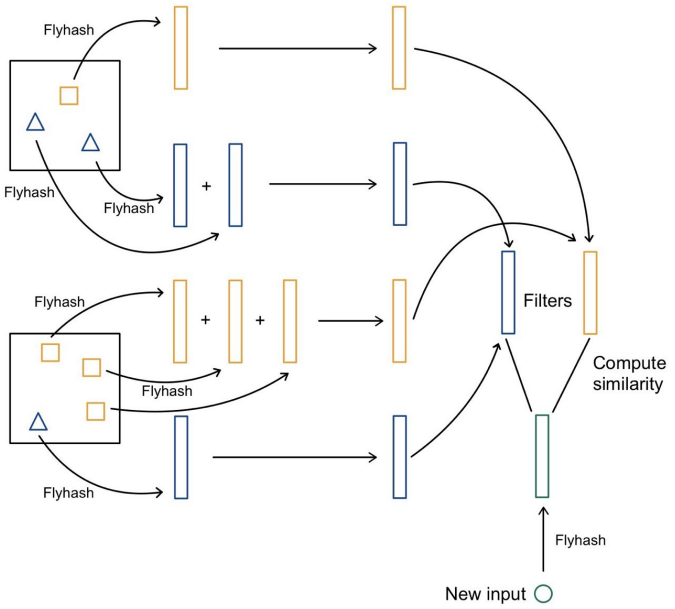


Fig. 2. Illustration of the FlyNN scheme to perform approximate k-Nearest Neighbors classification using FlyHash [RS22]. Each party, here represented by a square, hashes the data point it owns for each class (here orange squares and blue triangles) and sums them into a single hash vector. The parties then combine their hash vectors to obtain a single hash vector for each class across the entire dataset, called *filter*. The class of a new data point is then determined by the filter that has the highest similarity to the new data point's hash vector.

proximation of the k-nearest neighbor classification algorithm in the federated setting. FlyNN has in particular the advantage of being usable in the context of one-shot federated learning, where the communication among clients is restricted to a single round, and to be readily combinable with differential privacy schemes [RS22].

c) Speeding up neural network training via Locality-sensitive Hashing: An important application of LSH schemes has recently been provided in [DMZS21], in which such schemes are leveraged to speed up training and inference of large artificial neural network architectures on the CPU, based on the empirical observation that only a small fraction of neurons is activated per layer. A simplistic diagram of the SLIDE framework proposed in [DMZS21] is outlined in Fig. 3. The purpose of the present work is not only to increase the efficiency of the FlyHash algorithm as an alternative that can be leveraged in SLIDE, but also to make it possible to leverage the aforementioned framework to speed up training on the GPU.

III. Description of the Algorithms

A. FlyHash

Formally, the FlyHash algorithm takes as input a vector in $\mathbb{R}^d$ and returns its hash, which is a binary vector of length N (the *hashlength* parameter). The algorithm has two main parts, a projection and a winner-take-all binarization part.

The projection matrix is a random binary matrix M of size $N \times d$ with a fixed number s (the *projection* parameter) of ones

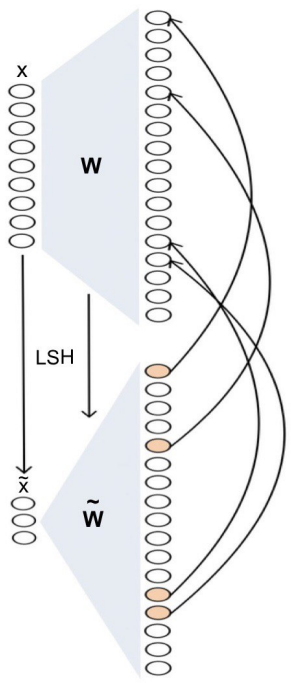


Fig. 3. Diagram illustrating the core idea of the SLIDE framework, in which activating neurons are predicted by computing the scalar product between the projection of the input $\tilde{x}$ and the projection of the layer weight matrix $\tilde{W}$ via a Locality-sensitive Hashing scheme. The actual input of the neurons that are predicted to activate is then computed via the original input x and the corresponding rows of the original weight matrix W.

in each row. The first part of the algorithm is the multiplication between M and the input vector.

The winner-take-all binarization is then applied to the outcome of the previous step, which is in $\mathbb{R}^N$, and transformed into a $\{0,1\}^N$ vector by setting the k highest entries to one and the others to zero. The parameter k is also called *number of winners* parameter.

Algorithm 1 contains the FlyHash pseudocode. The WTA function is explained in detail in the next section.

Algorithm 1 FlyHash

Input: $X \in \mathbb{R}^{d\times b}$, $M \in \{S \in \{0,1\}^{N\times d} :$ each row of S contains s ones$\}$, $k \in [1, N]$
Output: $X \in \{0,1\}^{N\times b}$
$A = M \times X$
return WTA(A, k)

B. A parallelized winner-take-all

We implement the FlyHash algorithm on the GPU to process large amounts of data. Our main contribution is a parallelized winner-take-all binarization algorithm that, rather than taking as input a single vector as mentioned before, processes a batch of vectors in a matrix X of size $N \times b$, where b is the batch size. The binarization step is thus applied to each column simultaneously. More specifically, we perform a parallel binary search for the values that, when used to threshold the respective columns, give the desired number of ones k.

Algorithm 2 contains the winner-take-all pseudocode. It starts by computing, for each column, the lower bound lb and the upper bound ub of the search interval by taking, respectively, the minimum and the maximum with a small margin $\varepsilon > 0$ to allow for strict inequalities. It then calculates the middle value mid and updates the extremes according to the current number of ones tot (corresponding to the number of values greater than mid): if they are greater than k, we increase the lower bound of the interval by setting it equal to the middle value; instead, if they are less than k, we decrease the upper bound to be equal to the middle value. The process is repeated a given number of times, which is at most 278 for single-precision floats, but in practice it can be set to 64 if a small fraction of erroneous entries can be tolerated (for example, the average fraction of erroneous entries caused by such a limitation is around 2.095×10^{-7} when the output is of size 20000×5000).

Algorithm 2 Winner-take-all (WTA). Functions preceded or followed by a dot (Julia's broadcasting operator) are applied element-wise.

Input: $X \in \mathbb{R}^{N\times b}$, $k \in [1, N]$
Output: $X \in \{0,1\}^{N\times b}$
$lb = \texttt{minimum}(X,\ \text{dims} = 1). - \varepsilon$
$ub = \texttt{maximum}(X,\ \text{dims} = 1). + \varepsilon$
$mid = (lb. + ub)./2$
for _ in 1 : 64 **do**
$tot = \texttt{count}(X. > mid,\ \text{dims} = 1)$
$lb = \textbf{ifelse}.(tot. > k, mid, lb)$
$ub = \textbf{ifelse}.(tot. < k, mid, ub)$
$mid = (lb. + ub)./2$
end for
return $X. > mid$

IV. Experiments

In our experiments, we compare the performance of the FlyHash algorithm on an `Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz` CPU and a `NVIDIA Quadro RTX 8000` GPU with `CUDA version: 11.8`, focusing on the processing of large amounts of data. We implemented the algorithms in the Julia programming language, which is now one of the most popular languages for scientific computing, and we relied on CUDA.jl package for the GPU part. On the algorithm engineering side, some optimizations have been made to speed up the process, such as pre-allocating variables for the GPU version. As for the CPU implementation, it uses efficient partial sorting algorithms to select the k winners [RS22]. Our code is available in the following Github repository: https://github.com/AInnervate/flyhash.jl, along with the code to replicate the experiments explained in more detail below.

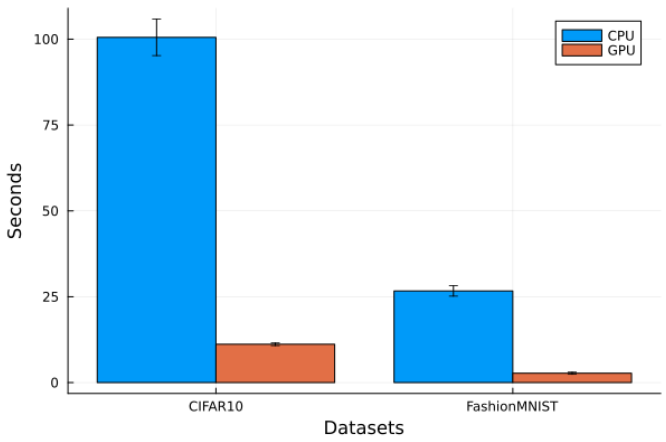


Fig. 4. Comparison on two popular datasets. Hash factor parameter is set to $h = 32$.

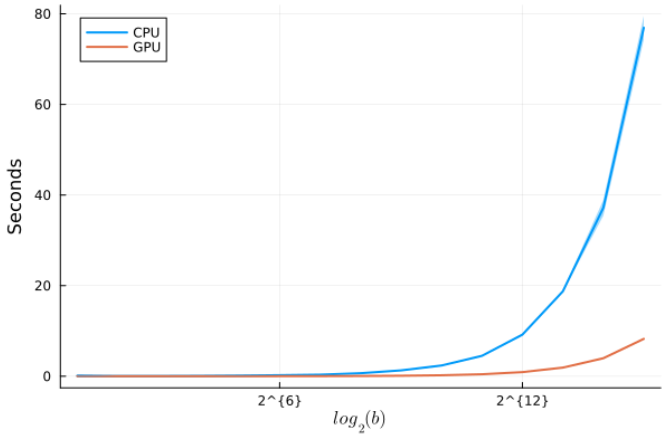


Fig. 5. Comparison as batchsize increases with fixed hash factor $h = 128$ and input dimension $d = 1024$.

First, we test the speed of the two versions of the algorithm on well-known datasets taken from the Machine Learning Datasets Julia library MLDatasets.jl. One of the two datasets we examine is the FashionMNIST dataset, which is a collection of 60000 greyscale images with a size of 28×28. We therefore transform each image into a vector of dimension $d = 28 \times 28 = 784$ and collect them in a matrix of dimension 784×60000. Then we pass it as input to the FlyHash algorithm, choosing the following parameters according to previous work [SDN17]: the hash length N is equal to the hash factor $h = 32$ multiplied by the input dimension d, the projection parameter s is set to 5% of the input dimension d and the number of winners k is set to 5% of the hash length N. The same configuration is used to compute the time to process the CIFAR10 dataset, which is a collection of 50000 coloured images with a size of 32×32, so the input dimension of the matrix is $d \times b$, where $d = 32 \times 32 \times 3 = 3072$ (the factor 3 comes from the fact that the images are coloured) and the batchsize is $b = 50000$.

In addition to those datasets, we perform experiments on synthetic data, where each entry is uniformly sampled in [0,1]. To understand how the two architectures behave when dealing with large amounts of data, we run the code varying the batchsize b in the range $\{2^1, ..., 2^{15}\}$ and fixing the other parameters: the projection parameter and number of winners are as above, the hash factor is set to $h = 128$ and the input dimension is $d = 1024$.

The results are shown in Figure 4 and Figure 5. They are obtained by averaging 10 independent runs after first running the code without taking the time to avoid Julia's just-in-time compilation overhead. Both plots show low standard deviation values, in the first case as bars and in the second as shadows.

V. Conclusions

In this work, we proposed a GPU implementation of the famous FlyHash locality sensitive algorithm. Experiments show that our GPU version can run one order of magnitude faster than the best CPU version, thus allowing to speed up the use of the FlyHash algorithm in settings where GPUs are more easily available than dozens of CPU cores. In future work, we expect that the GPU code can be further improved in terms of performance by writing a lower-level CUDA kernel.

Important directions consist in exploring the implications of our implementations for the use of FlyHash in real-world applications, such as the ones mentioned in Section II.

References

[Cha02] Moses Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing*. ACM, 2002.

[Che17] Yanqing Chen. Mechanisms of winner-take-all and group selection in neuronal spiking networks. *Frontiers in computational neuroscience*, 2017.

[DMZS21] Shabnam Daghaghi, Nicholas Meisburger, Mengnan Zhao, and Anshumali Shrivastava. Accelerating SLIDE Deep Learning on Modern CPUs: Vectorization, Quantizations, Memory Optimizations, and More. *Proceedings of Machine Learning and Systems*, 3:156–166, March 2021.

[DSSN18] Sanjoy Dasgupta, Timothy C. Sheehan, Charles F. Stevens, and Saket Navlakha. A neural data structure for novelty detection. *Proceedings of the National Academy of Sciences*, 2018.

[IM98] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98. Association for Computing Machinery, 1998.

[MVSG+09] O. Moslah, A. Valles-Such, V. Guitteny, S. Couvet, and S. Philipp-Foliguet. Accelerated multi-view stereo using parallel processing capabil-

ities of the gpus. In *2009 3DTV Conference*, 2009.

[PVM+20] Christos Papadimitriou, Santosh Vempala, Daniel Mitropolsky, Michael Collins, and Wolfgang Maass. Brain computation by assemblies of neurons. *Proceedings of the National Academy of Sciences of the United States of America*, 2020.

[RS22] Parikshit Ram and Kaushik Sinha. Federated nearest neighbor classification with a colony of fruit-flies. In *Thirty-Sixth AAAI Conference on Artificial Intelligence*. AAAI Press, 2022.

[SDN17] Charles F. Stevens Sanjoy Dasgupta and Saket Navlakha. A neural algorithm for a fundamental computing problem. *Science*, 2017.

[SR21] Kaushik Sinha and Parikshit Ram. Fruit-fly inspired neighborhood encoding for classification. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 2021.

[YSRL11] Jay Yagnik, Dennis Strelow, David A. Ross, and Ruei-sung Lin. The power of comparative reasoning. In *2011 International Conference on Computer Vision*, 2011.

Securing Intellectual Property in Federated Learning

Mohammed Lansari*†, Reda Bellafqira*, Katarzyna Kapusta†,
Vincent Thouvenot†, Olivier Bettan†, Gouenou Coatrieux*

* Inserm UMR 1101, IMT Atlantique
Brest, France
name.surname@imt-atlantique.fr

† ThereSIS, Thales SIX GTS
Palaiseau, France
name.surname@thalesgroup.com

***Abstract*—Federated Learning (FL) is a technique that allows multiple participants to collaboratively train a Deep Neural Network (DNN) without the need to centralize their data and therefore comes with privacy-preserving properties making it attractive for application in sensitive contexts. However, it requires sharing participant models during the training process which makes them vulnerable to theft or unauthorized distribution by malicious actors. To address the issue of ownership rights protection in the context of Machine Learning (ML), DNN Watermarking methods have been developed during the last five years. Most existing works have focused on watermarking in a centralized manner, but only a few methods have been designed for FL and its unique constraints. In this paper, we provide an overview of recent advancements in Federated Learning watermarking, shedding light on the new challenges and opportunities that arise in this field.**

***Index Terms*—ML Watermarking, Federated Learning, Intellectual Property, Security.**

I. Introduction

Sophisticated ML models require good and non-synthetic data that are often hard and costly to acquire. Government regulations, such as the European General Data Protection Regulation, have made the training process even harder by requiring to preserve privacy of the data subjects. Thus, companies and institutions have to face the issue of lack of data on the one hand and the complexity of usage of sensitive data on the other. Indeed, very often the data exists but is distributed over multiple locations and/or belongs to different owners. Centralizing it would be very costly in terms of bandwidth. Moreover, it is almost impossible if the decentralized data is private or classified.

Federated Learning [1] is a promising solution that allows multiple data owners (or distributed agents belonging to one data owner) to train a global Deep Neural Network (DNN) without directly sharing their data. It opens new possibilities in domains such as health or military, as it allows to train a model over multiple distributed datasets without centralizing the local data. Combined with privacy-preserving techniques, such as Homomorphic Encryption, Multi-Party Computation, or Differential Privacy, it can ensure a high level of protection of sensitive data. In the civil domain, it may be applied by a consortium of hospitals, where each of the hospitals possesses its own data that is impossible to share due to regulation constraints. In the military domain, it could find application for instance in the case of predictive maintenance, where multiple cooperating parties would like to benefit from the information collected by others but fear about revealing classified information.

From the point of view of ownership rights protection, the complex context combining both AI and collaboration of owners and users raises new challenges. ML watermarking techniques have been proposed since 2018. They enable to identify a ML model and thus can be applied for model theft detection. However, the majority of them address the problem of local training of a single model. The collaborative setting of FL raises new challenges in the context of ownership protection. How to identify that a participant contributed to the training of a model resulting from FL? How to be sure that the final model will not be misused by the aggregation party that coordinates the FL? These issues may slow down parties from joining the learning consortium.

II. Outline

We start with a brief presentation of the FL and DNN watermarking concepts in Section III. Then, we proceed with a definition of watermarking for FL in Section IV followed by an overview of the six existing works combining FL and DNN. In Section V, we analyze the different elements that have to be taken into consideration while designing a watermarking scheme for collaborative training settings. We identify the unsolved challenges and thus give an insight into possible future works.

III. Background

In this section, we remind the definition of FL and give a brief overview of its different existing settings. Then we introduce DNN watermarking.

A. Federated Learning

FL is a ML setting where $K \in \mathbb{N}^*$ entities called clients collaborate to train a global model M_G while keeping the training data $D_C = \{D_{C_i}\}_{i=1}^{K}$ decentralized [2].

The most popular framework is called Client-Server FL. In this setting, the server train M_G by receiving and aggregating the client models weights $W_C = \{W_{C_i}\}_{i=1}^{K}$ (see e.g. [3]). However, the presence of the server is not mandatory to perform FL. In a decentralized setting, clients can perform FL without the supervision of a server. BrainTorrent [4] proposes a server-less and peer-to-peer Federated framework. In this solution, a random client is selected to be the aggregator. Then, it checks if other clients have an updated version of the model. If yes, they send it to the aggregator who performs an averaging of the model weights. Then it updates its own model with the previous result.

The FL setting is also characterized by the features partition. We distinguish three types of partition: horizontal, vertical, and hybrid FL. In horizontal FL, all clients have the same features but not the same samples. On the other hand, vertical FL assumes that all clients have the same samples but not the same features. Finally, hybrid FL supposes that samples and features are different from one client to another.

In contrast with previous settings, Split-Learning [5] consists of splitting the DNN between the server and the clients. Many configurations are possible but the most common for client privacy is the U-shaped configuration. The aim is that each client has the first and last layers of the DNN and the server has the rest of the layers. In such a way, clients perform the forward/backward pass keeping their data (inputs and labels) private, and send/receive only the activations/gradients to update the whole model.

B. DNN Watermarking

DNN watermarking is a promising solution for the ownership protection of ML models. Inspired by image watermarking, it consists of introducing a secret change into the model parameters or behavior during its training, in order to enable its identification in the future. As image watermarking, DNN watermarking must respect some requirements to be effective for IP protection. Table I summarizes these requirements. In general, a watermarking technique needs to preserve the performances on the main task (**Fidelity**), while providing a large insertion **Capacity** (enabling multiple verifications) and a strong **Robustness** against watermarking removal techniques [6].

DNN Watermarking can be distinguished into two types of techniques: White-Box [7] [8] [9] [10] and Black-Box [8] [11] [12] [13] [14] watermarking. Each technique is defined by the type of access to the model during the verification process.

In the White-Box setting, we assume that the owner will have full access to the model (architecture, weights, activations, etc.). In this way, to insert a watermark into a DNN, the owner will hide a vector of bits b into the model's parameters or activations. One of the first proposed method in [7], uses a regularization term to embed b with a secret key S into the model's parameters. Several techniques have been published to meet the associated challenges. DeepSigns [8] proposes to use the probability density function of the output for selected layers to embed the information. Tartaglione *et al.* [9] defines a strategy that consists in embedding the watermark before the training and then training the model while adding a constraint that penalizes the model for small perturbations on the watermarked parameters.

On the other hand, Black-Box setting assumes that the owner can perform the verification process only through an API: he can interact with the model only by giving inputs and receiving associated predictions. Knowing that the owner watermarks the model by changing its behavior. The common technique consists training of the model using a trigger set $T = (X_i, Y_i)_{i=1}$, which is composed of crafted inputs X_i with their associated outputs Y_i [11]. Zhang *et al.* [12] propose to use the same technique but with different types of inputs. In addition to unrelated images, they use training examples with two types of trigger: random noise and a textual pattern. The trigger set can also be built using adversarial examples [13] or a filter as a mapping function [14].

To evaluate the performances of the watermarking embedding, White-Box setting relies on the Binary-Error-Rate between the reconstructed message $\tilde{b}$ and the original one b. In Black-Box watermarking, we evaluate the accuracy of the model on the trigger set T.

IV. Watermarking for Federated Learning

In this section, we introduce and define what is watermarking for FL including the different possible scenarios. Then, we formulate requirements for watermarking deployed in a FL context. Finally, we analyze the six state-of-the-art methods.

A. Definition

In centralized DNN watermarking, the goal is to simply prove the model's ownership after the training process. In FL, ownership rights protection becomes a more complex problem due to the presence of multiple participants and multiple exchanges between them that have to be taken into account during the threat model formulation. To illustrate this issue, [15] shows that existing methods can naively be applied to FL in two manners. The first one consists of watermarking the model after the training. For example, by using fine-tuning to embed the watermark into the model. Without taking the fidelity requirement into account, any participant who receives the model (client or server) can steal the DNN before the last round. The second way is to embed the watermark before the training. Even if the watermark will resist during the first rounds, it will be removed after several aggregation rounds. Thus, it is important to design a specific watermarking technique for FL that will be persistent from the first round to the model deployment.

We define Watermarking for FL as the process for a participant or multiple participants to insert a watermark into the shared model.

Following the client-server FL framework, the first question is to determine which part of the federation can watermark the model. Is the server more to be trusted since it manages the federation? Or the clients since their data are used? During this study, we distinguish four watermarking scenarios for

Fidelity	The watermarked model needs to have the same performances compared to the model without watermark
Capacity	The capacity of a technique to embed multiple watermarks
Generality	The capacity of a watermarking technique to be applied independently of the architecture of the model
Efficiency	The performance cost generated by the embedding and verification process of the watermarking
Robustness	The capacity to resist against attacks aiming at removing the watermark
Secrecy	The watermark should be secret and undetectable

TABLE I: DNN Watermarking requirements

centralized FL, which we illustrate in Figure 1 according to who is watermarking the model :

- (**S$_1$**) **Server** : The server is in charge of watermarking the global model.
- (**S$_2$**) **Clients** : One or multiple clients among the federation watermark their updates to spread that are embedded into the global model.
- (**S$_3$**) **Server and clients** : The server and the clients collaborate to watermark the global model together.
- (**S$_4$**) **Clients in a decentralized context** : Only clients collaborate to watermark the global model together (decentralized FL).

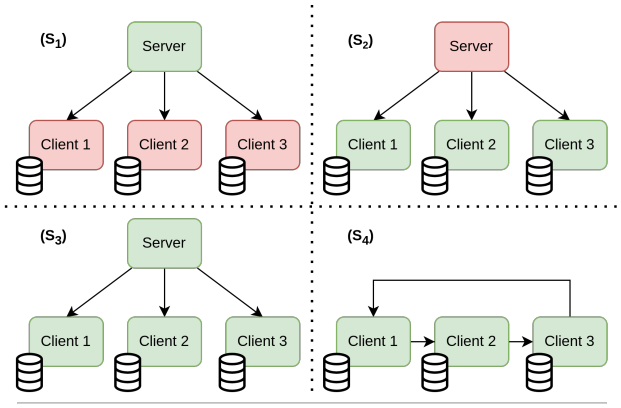


Fig. 1: An example of each possible scenario of watermarking in FL with one server and three clients. Green rectangles are the participants who follow the same watermarking procedure together. Red rectangles are those who are not enrolled in the watermarking procedure.

All watermarking requirements defined in Table I are also true in the federated context. However, due to the new constraints and the extension to several participants, we can add precision to three of them:

1) **Capacity :** When multiple clients want to insert their own message b_{C_i}, the watermarking technique needs to avoid possible conflict between the different inserted b_{C_i}. The number of bits needs to be enough.
2) **Generality :** In a real FL scenario, many additional mechanisms are added for security and privacy such as robust aggregation functions (Section V-B) or Differential Privacy [16] (Section V-E). The watermarking technique must be applied independently to these mechanisms.
3) **Efficiency :** The cost generated by the embedding process is more crucial in FL. For example, in a cross-device architecture, clients have low computation power and they cannot perform many operations. The watermarking techniques must take this parameter into account.

B. Related works

1) WAFFLE: WAFFLE [15] is the first DNN Watermarking technique for FL. In this solution, the server embeds the watermark (**S$_1$**) using a black-box watermarking technique using a trigger set. Any trigger set that does not need any knowledge concerning the clients' data can be used but the authors present a specific set that is more suitable for FL: the WAFFLEPattern. WAFFLEPattern is defined as a set of images containing random patterns with a noisy background. Basically, the server will embed the watermark into the global model using the two following functions :

- **PreTrain** : Train an initialized model with the trigger set until
- **ReTrain** : Fine-tune (using FTAL) the model with the trigger set until

PreTrain is used to embed the watermark in the model before the first round. During each round, after the aggregation process, the server uses **ReTrain** to re-embed the watermark into the model.

2) FedIPR: FedIPR [17] is both a Black-Box and White-Box technique. This technique allows all clients to embed their own watermark in the global model (**S$_2$**) without sharing secret information. Each watermark can be described as follow :

- **Black-Box Watermark** : Each client generates a trigger set using Projected Gradient Descent technique in a small CNN trained with the local data.
- **White-Box Watermark** : Each client generates a random secret matrix and a location in the Batch-Normalisation layers to embed its message.

Both White-Box and Black-box watermarks are inserted using an additional loss during the local training. For **Black-Box Watermark**, the loss is exactly the same as the loss used for the main task but with a batch of the trigger set as input. For the **White-Box Watermark**, the loss used is a Hinge-like loss between the original message and the reconstructed message.

3) FedTracker: FedTracker [18] is a watermarking technique that allows the server to embed a global Black-box watermark (**S$_1$**) but also a White-box watermark specific to each client. Each watermark can be described as follow :

- **Global Black-Box Watermark** : A trigger set is generated using the WAFFLEPattern method [15].
- **White-Box Watermark** : Server generates a random secret matrix and a fingerprint for each client.

After the aggregation, the server embeds the **Global Black-Box Watermark** using the intuition of Continual Learning [19] to avoid forgetting the main task. Then, for the **White-Box Watermark**, the loss used is a Hinge-like loss between the original message b and the reconstructed message $\tilde{b}$.

4) Client-side Black-box watermarking: Liu *et al.* [20] propose a client-side Black-box watermarking scheme (**S_2**). This technique is designed to embed a watermark only from one client. The latter creates a trigger set composed of Gaussian noise images with a given label as a trigger set. Then, the client's model will over-fit with this set like in [11]. To tackle the fact that this particular client will probably not be selected at each round, the authors introduce a scaling factor

$$\lambda = \frac{N}{n},$$

where N is the number of clients and n is the number of clients selected at each round. The client will then send its model weights multiplied by λ. According to the authors, this will be approximately equivalent to the case that this client is selected every iteration and the watermark will be embedded more easily.

5) Merkle-Sign: Merkle-Sign [21] is a framework focusing on ownership verification in a collaborative Clients-Server setting (**S_3**). The authors propose a public verification protocol that uses the Merkle-tree [22]. In this framework, the server use at each round an embedding function to insert two identity information (i.e keys) into M_G : one that identify the server and the other one the client that will receives the model. In parallel, the server uploads the tuple of keys and the tuple of verification function (which are generated by the watermarking embedding function) into a Merkle-tree with the recording time. At the final round, the server embeds also all keys generated by the clients into M_G and updates the Merkle-tree. This framework is also compatible in a Peer-to-Peer context. The associated Black-Box watermarking schema relies on the training of an Auto-Encoder [23] (AE) for each client. Then the server averages the received AE to obtain a final AE from which it can generate a trigger set using the keys as input for the decoder part.

6) FedRight: FedRight [24] is a solution for the server to fingerprint the model in the FL framework (**S_3**). DNN fingerprinting is a process in which instead of embedding a watermark in the model, we extract a fingerprint to identify this model [25]. To do so, the server generates adversarial examples from a set of inputs (key samples). Then the server extracts the probability distribution of each prediction and feeds it to a detector with the key samples target. Then, during the verification process, this detector is used to predict whether the corresponding model is the good one or not.

V. Discussion

In this section, we identify and discuss specific challenges related to watermarking in FL. In particular, we evaluate how existing methods deal or not with these new challenges.

A. Black-Box watermarking in the Server side

Black-Box watermarking consists in changing the behavior of the model. To do so, most methods let the model overfit on the trigger set by adding a regularization term in the loss function. Usual DNN watermarking techniques can easily be applied in **S_2**, **S_3** and **S_4**. However, it is not so easy in **S_1**. When the client C_k wants to watermark its model M_{C_k}, he can rely on two things :

1) Have an access to its private dataset D_{C_k}
2) Train the model on both the main task dataset D_{C_k} and its trigger set T_{C_k} at the same time

A large number of Black-Box watermarking techniques need to build T_{C_k} using $D_{C_k} = (X_i, Y_i)_{i=1}$ (as discussed in III-B). Using such techniques is motivated by the fact that training the model from these datasets is a multi-task learning. Building T_{C_k} from D_{C_k} helps to reduce the negative impact on learning from two domains. Moreover, learning these two tasks together avoids "catastrophic forgetting" [26].

In **S_1**, since the server does not have its own dataset, it cannot perform such type of watermarking. The choice is limited by using unrelated or noise-based inputs as WafflePattern [15].

The problem is that this limitation leads to an exposure to evasion attacks. During the Black-Box verification process, the owner will ask the suspicious Application Programming Interface (API) that possibly contains his DNN. However, the attacker may evade this verification using a query detector [27]. Since the trigger-set is built using images that are qualified to be Out-Of-Distribution (OOD), this implies an easier detection for the attacker [28]. WAFFLE [15] authors confirm the intuition that the performances of such detector depends a lot on the data quantity and capacities of the attacker.

B. Aggregation functions

The most common aggregation function is `FedAvg` [1] which consists on averaging clients' weights after they perform multiple epochs on mini-batches. Each client weight matrix is multiplied by a scaling factor defined as $\frac{n_{C_k}}{n}$ where n_{C_k} is the number of samples in D_{C_k} and $n = \sum_k^K n_{C_k}$. Many aggregation functions emerged to meet various challenges in FL. Since the clients do not necessarily know which aggregation function the server is using, the proposed methods must be independent of this parameter.

For the Byzantine-attacks problem in which one or multiple clients try to disturb the FL process. These attacks can be simple noise weights or complex label-flipping backdoors. To

leverage this problem, multiple aggregation functions appear to select only benign updates such as `Krum` [29] `Trim-mean` or `Bulyan` [30]. Since clients' watermarking techniques are sensitive to the embed message b and the trigger set T, they keep their updates far from each other. A part of updates can be rejected for this reason if we use defensive aggregation techniques. As an example, FedIPR shows that the **White-Box Watermark** results are similar to `FedAvg` with a detection rate of 97.5% using `Trim-mean`. However, the **Black-Box Watermark** reaches only 63.25% of the watermark detection rate at the end of the FL process. Even if this score is enough to detect plagiarism, using a defensive aggregation function has a huge impact on the watermark. Liu et at. [20] have not tested yet their solution with a defensive aggregation function but we can guess that multiplying weights by a so big scaling factor λ can be easy to detect for `Krum` as a Byzantine attack as shown in similar example [31].

Another problem is that `FedAvg` performs well when the data are statistically homogeneously distributed among the clients. However, in real use cases, data are heterogeneous which may lead to difficulty for the model to converge to the global minimum or diverge using `FedAvg`. Existing watermarking techniques for FL have not evaluated methods that tackle this problem such as `FedProx` [32], `FedNova` [33] or `SCAFFOLD` [34]. If we want to use the proposed solution in a real Secure FL framework, these methods need to be tested in such a context.

C. Client selection

For communication efficiency and when the number of clients is too high, we introduce a client selection. To do so, we simply randomly select cK clients with $c \in]0, 1[$. This simple mechanism can have a big effect on the watermarking. In ($\mathbf{S_1}$), this effect should be insignificant, since the server embeds the watermark at each round regardless of cK. Unfortunately, WAFFLE and FedTracker have not tested this effect. On the other hand, ($\mathbf{S_2}$) is more able to be sensitive to the client selection process. Since each client wants to insert its watermark, the watermark of not selected clients risks to be degraded or removed in the global model. Authors of FedIPR show that for $c > 0.2$ the detection rate for both White-Box and Black-Box watermark is near to 100%. When $c \leq 0.2$ the detection rate associated with a feature-based watermark is still near 100%. However, the detection rate for the backdoor falls to 62%.

D. Cross-device setting

All proposed papers are treating the case in which we have a small amount of clients. The worse scenario is tested in Merkle-Sign in which 200 clients are enrolled in the federation. However, there is no solution that takes into account the cross-device setting. In this setting, a large number of clients (up to 10^{10} devices), are enrolled in the FL procedure [2]. These clients are not always reachable and they have a low dedicated computational power which is defined as a performance condition by the authors of WAFFLE.

In the Black-Box setting, the problem can come from the low computation power that does not allow the client to perform more computations to increase the batch size using trigger-set methods. For the White-Box setting, the bottleneck would be the **Capacity** as mentioned in Section IV-A. In particular in cases where the proposed methods are tested using Normalization layers such as FedIPR or FedTracker which limits the overall embedding capacity. It leads to a difficulty for each client to embed its vector b without conflicting in face of other clients' watermarks.

E. Differential Privacy

Since FL keeps the client's data private, DNNs are shared between the server and the clients or directly between clients. The model himself can give to an attacker private information in such a way that he can identify the presence of an exact data point (such an attack is called membership inference) [35]. To tackle this problem, we can use Differential Privacy (DP) which is a strong standard for providing privacy guarantees for algorithms operating on aggregate databases [16]. In FL, a usual DP technique consists of adding a Gaussian random noise to the gradients sent to the aggregator. Among related works, FedIPR is the only article for which the proposed method is tested with a DP mechanism. The presented results show that adding a small noise to the weights does not affect a lot the watermarking accuracy.

F. Homomorphic Encryption

In Client-Server FL, the server has access to all client updates. Then, it can use this privilege to try to learn information about the private datasets. A cryptographic solution to prevent such misuse from the server is, for the clients, to protect the model using Homomorphic Encryption (HE). By using this technique, clients can easily cipher their updates using public keys. Then the server will perform the aggregation, in general using `FedAvg`, in the cipher space. Then, when clients receive the updated global model, they use their private key to decipher the model weights.

In the context of watermarking, the use of HE fits perfectly for ($\mathbf{S_2}$) since the clients can access their model weights to watermark the model. However, so far no solution has been provided for ($\mathbf{S_1}$) given the fact that the server cannot decipher the model weights to perform watermarking embedding.

G. Watermarking for Non Client-Server framework

Decentralized FL ($\mathbf{S_2}$) is an interesting framework in which clients do not need a server to perform the model aggregation. The proposed methods seem to be applicable to this setting since watermarking the model from the client side does not require the server. However, Merkle-Sign is the unique solution that extends to the decentralized setting ($\mathbf{S_4}$). We can also cite Split-Learning in which the server has a part of the network and clients have another part. Performing White-Box watermarking as in [7] can be more difficult for the server or clients. In both cases, they have access to a part of the model weights that can be arbitrarily small. In the U-shape

Split Learning architecture, the server has only the middle part of the model and the client has the first and last layers. In this setting, performing a Black-Box watermarking on the server side is hard since it cannot use its inputs and labels on the model.

H. Attacks from clients and server

When we analyze ($\mathbf{S_1}$) and ($\mathbf{S_2}$) scenarios, we can see that each one has different parameters to play with whether for watermarking or disrupt it. The server can control the selected participants or how to aggregate the weights. It has also sometimes a clear view of clients' weights at each round. However, it does not have data and it cannot fully control if clients are strictly following the training process. On the other hand, clients have their private dataset and they can send the weights that they want. Nevertheless, they have no control over what happens with their updates in the server level.

If the server wants to avoid a subset of clients to watermark the model, it can use methods proposed for Byzantine attacks [36] detection. In particular, attacks that consist of multiplying the weights by a scaling factor to replace or have a bigger impact in the global model are easy to detect [31]. The proposed method by Liu *et al.* [20] is then easily removable and the global model will not be watermarked. A solution to catch backdoor-ed models was presented in [37] [38]. Then all proposed solutions that rely on a backdoor-based watermarking can be rejected.

Clients can also try to disturb the watermarking process. FedIPR ($\mathbf{S_2}$) [17] authors present the free-riders problem in which some clients do not contribute to the training of M_G and the watermarking process. Even if with their solution, they have no important impact on the watermarking, no testing has yet been done on ($\mathbf{S_1}$). Another attack that is specific to FL as described in FedRight [24] and WAFFLE [15] consists of the fact that multiple clients will use their models and private datasets. As mentioned in Section V-A, evasion attack works better when using multiple datasets to train the detector. But it is also possible to fine-tune the model with the combined dataset.

VI. Conclusion

Watermarking for FL is taking great interest since classical DNN watermarking cannot be naively applied in a collaborative context. In particular, constraints such as data distribution, new distributed threat models, and additional security mechanisms have to be taken into account while designing an efficient solution for collaborative ML watermarking. This paper provides a comprehensive overview of existing methods and exposes the different problems they face off. Several of the analyzed methods have tried their model against each Secure FL mechanisms separately. Unfortunately, none of them has tried to test their solutions in a complete and realistic Secure FL framework, including defensive aggregation functions, non I.I.D data, DP, and HE in the same experiment.

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[2] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[3] H. Brendan McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," *arXiv e-prints*, pp. arXiv–1602, 2016.

[4] A. G. Roy, S. Siddiqui, S. Pölsterl, N. Navab, and C. Wachinger, "Braintorrent: A peer-to-peer environment for decentralized federated learning," *arXiv preprint arXiv:1905.06731*, 2019.

[5] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar, "Split learning for health: Distributed deep learning without sharing raw patient data," *arXiv preprint arXiv:1812.00564*, 2018.

[6] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 707–723.

[7] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, 2017, pp. 269–277.

[8] B. Darvish Rouhani, H. Chen, and F. Koushanfar, "Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2019, pp. 485–497.

[9] E. Tartaglione, M. Grangetto, D. Cavagnino, and M. Botta, "Delving in the loss landscape to embed robust watermarks into neural networks," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 1243–1250.

[10] Y. Li, B. Tondi, and M. Barni, "Spread-transform dither modulation watermarking of deep neural network," *Journal of Information Security and Applications*, vol. 63, p. 103004, 2021.

[11] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *27th USENIX Security Symposium*, 2018, pp. 1615–1631.

[12] J. Zhang, Z. Gu, J. Jang, H. Wu, M. P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, pp. 159–172.

[13] E. Le Merrer, P. Perez, and G. Trédan, "Adversarial frontier stitching for remote neural network watermarking," *Neural Computing and Applications*, vol. 32, pp. 9233–9244, 2020.

[14] J. Guo and M. Potkonjak, "Watermarking deep neural networks for embedded systems," in *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2018, pp. 1–8.

[15] B. G. Tekgul, Y. Xia, S. Marchal, and N. Asokan, "Waffle: Watermarking in federated learning," in *2021 40th International Symposium on Reliable Distributed Systems (SRDS)*. IEEE, 2021, pp. 310–320.

[16] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 308–318.

[17] B. Li, L. Fan, H. Gu, J. Li, and Q. Yang, "Fedipr: Ownership verification for federated deep neural network models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[18] S. Shao, W. Yang, H. Gu, J. Lou, Z. Qin, L. Fan, Q. Yang, and K. Ren, "Fedtracker: Furnishing ownership verification and traceability for federated learning model," *arXiv preprint arXiv:2211.07160*, 2022.

[19] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars, "A continual learning survey: Defying forgetting in classification tasks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 7, pp. 3366–3385, 2021.

[20] X. Liu, S. Shao, Y. Yang, K. Wu, W. Yang, and H. Fang, "Secure federated learning model verification: A client-side backdoor triggered watermarking scheme," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2021, pp. 2414–2419.

[21] F.-Q. Li, S.-L. Wang, and A. W.-C. Liew, "Towards practical watermark for deep neural networks in federated learning," *arXiv preprint arXiv:2105.03167*, 2021.

[22] G. Becker, "Merkle signature schemes, merkle trees and their cryptanalysis," *Ruhr-University Bochum, Tech. Rep*, vol. 12, p. 19, 2008.

[23] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," *arXiv preprint arXiv:2003.05991*, 2020.

[24] J. Chen, M. Li, and H. Zheng, "Fedright: An effective model copyright protection for federated learning," *arXiv preprint arXiv:2303.10399*, 2023.

[25] X. Cao, J. Jia, and N. Z. Gong, "Ipguard: Protecting intellectual property of deep neural networks via fingerprinting the classification boundary," in *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, 2021, pp. 14–25.

[26] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[27] D. Hitaj and L. V. Mancini, "Have you stolen my model? evasion attacks against deep neural network watermarking techniques," *arXiv preprint arXiv:1809.00615*, 2018.

[28] J. Yang, K. Zhou, Y. Li, and Z. Liu, "Generalized out-of-distribution detection: A survey," *arXiv preprint arXiv:2110.11334*, 2021.

[29] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," *Advances in neural information processing systems*, vol. 30, 2017.

[30] R. Guerraoui, S. Rouault *et al.*, "The hidden vulnerability of distributed learning in byzantium," in *International Conference on Machine Learning*. PMLR, 2018, pp. 3521–3530.

[31] Z. Gu and Y. Yang, "Detecting malicious model updates from federated learning on conditional variational autoencoder," in *2021 IEEE international parallel and distributed processing symposium (IPDPS)*. IEEE, 2021, pp. 671–680.

[32] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020.

[33] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *Advances in neural information processing systems*, vol. 33, pp. 7611–7623, 2020.

[34] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5132–5143.

[35] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *2019 IEEE symposium on security and privacy (SP)*. IEEE, 2019, pp. 691–706.

[36] J. Shi, W. Wan, S. Hu, J. Lu, and L. Y. Zhang, "Challenges and approaches for mitigating byzantine attacks in federated learning," in *2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2022, pp. 139–146.

[37] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I 25*. Springer, 2020, pp. 480–501.

[38] B. Xi, S. Li, J. Li, H. Liu, H. Liu, and H. Zhu, "Batfl: Backdoor detection on federated learning in e-health," in *2021 IEEE/ACM 29th International Symposium on Quality of Service (IWQOS)*. IEEE, 2021, pp. 1–10.

Détection et and identification radar de mini-drones à l'aide d'un réseau de neurone léger.

Jean-François DEGURSE[1], Pierrick RICHARD[1], Ronan GUILLAMET[1], Judie GUEGAN[1], Guillaume POINT[2], Ezequiel CENTOFANTI[3], Alain PEDEN[3], Vincent MULLER[4], Yves AUDIC[1]

[1] THALES DMS FRANCE, 10 Av. 1ère Dfl, 29200 Brest, France
[2] STELLANTIS, Rte de Gisy, 78140 Vélizy-Villacoublay, France
[3] IMT Atlantique, 655 Av. du Technopôle, 29280 Plouzané, France
[4] Drone-Act, 28 ter Pl. du Dr Jean Queinnec, 56140 Malestroit, France

Résumé – La détection et la classification d'aéronefs à voilure tournante est un enjeu majeur dans les domaines de la défense et de la sécurité. En particulier, la prolifération de drones multi-rotor de petite taille, modifiés pour des missions d'attaque ou d'observation sur des théâtres d'opérations militaires, représente une menace croissante. A partir de moyens compacts et embarqués, les méthodes les plus utilisées pour détecter ces menaces, et éventuellement reconnaitre, sont le systèmes optiques et acoustiques [1][2]. L'utilisation d'un radar en complément de ces systèmes peut améliorer les performances de détection et de classification. En effet, par rapport à une caméra optique, il est tout-temps et n'exploite pas sa résolution angulaire pour faire l'identification de la cible, et en comparaison d'un capteur acoustique, ses signaux évoluent dans un environnement moins bruité. Lorsque les contraintes SWAP (encombrement, poids, et consommation) n'est pas une contrainte forte, les radars très utilisés pour la détection de mini-drones [3][4]. En radar, pour classifier le drone en tant que tel, il est possible d'utiliser les signaux issus de sa signature micro-Doppler, qui rassemble les contributions de toutes les parties mobiles de l'aéronef, tel que les rotors. En fonction de la nature de la cible, cette signature peut être assez complexe mais peut être exploitée afin de détecter et identifier des drones à voilure tournante. Nous nous intéressons dans cet article, à l'utilisation d'un algorithme de réseau de neurones peu profond pour détecter et reconnaitre différents types de drones à partir d'un radar très compact. Afin de disposer d'une base de données suffisamment grande, et pour que ces données ne soient pas protégées, nous utiliserons un radar sur étagère à bas coût fonctionnant en mode CW.

Abstract – Rotorcraft detection and classification is a major issue for defense and security. In particular, the proliferation of small multi-rotor UAV, sometimes modified for attack or observation missions in military operation field represents a growing treat. Small and embeddable mini-UAV detection systems are usually based on optical and acoustic sensors [1][2]. The use of a radar in addition of these sensors can increase detection and classification performance. Indeed, when compared to optical camera, radar is not affected by weather condition and does not use its angular resolution to perform target identification, and when compared to acoustic sensors, relies on a less noisy environment. When SWAP (Size, Weight And Power) is not a strong constraint, radars are widely used for drone detection [3][4]. In radar, to perform drone recognition, it is possible to use the signal of its micro-Doppler signature, which gathers all contributions of moving part of the drone like rotors. In the paper, we focus on the use of a lightweight neural network algorithm to detect and identify different mini-UAV on small-sized radar data. In order to get a large amount of non-classified data, we chose to use a commercial off-the-shelf K-band radar in continuous wave mode.

1 Introduction

1.1 Contexte

Face à la prolifération des menaces, des systèmes radar très courte portée ont émergées afin de doter certains blindés de protection active. Ces radars ont pour but de détecter, localiser et pister les menaces (roquettes, missiles anti-blindés) arrivant à grande vitesse sur le porteur afin de déclencher une contre-mesure [5][6].

Les conflits récents ont montré que les mini-drones constituent une menace aussi importante en étant utilisés pour larguer des explosifs ou directement comme vecteur portant la munition jusqu'à l'impact. Par rapport aux menaces plus classiques, les vitesses d'approche sont plus faibles et les trajectoires sont moins linéaires, ce qui demande une adaptation du système de détection et des contre-mesures associées pour assurer leur interception.

Dans une première partie, nous détaillerons le modèle mathématique des signaux radars rétrodiffusés par un aéronef à voilure tournante, puis nous présenterons le protocole de mesure et d'enregistrement des données. Enfin nous présenterons les résultats en détection et en classification obtenus sur différents drones à l'aide d'un algorithme de machine learning nécessitant peu de ressources de calcul.

1.2 Modélisation des signaux radar d'une cible à voilure tournante

La signature radar d'un aéronef à voilure tournante peut se décomposer en 3 éléments [8] : le corps de l'aéronef, le hub du rotor et les pales du rotor.

Le corps de l'aéronef, également appelé cellule, qui possède une signature Doppler simple, dépendant de sa vitesse radiale relative par rapport au radar. La surface équivalente radar (SER) du corps de l'aéronef peut être modélisé de différentes façons, par exemple en prenant une loi de fluctuation Swerling 1. Le signal correspondant s'exprime donc de la manière suivante :

$$s_{corps} = \sqrt{\sigma_{corps}(t)}e^{i(\varphi_0+\varphi_{corps}(t)+2\pi f_D t)} \quad (1)$$

avec $\sigma_{corps}(t)$ la surface équivalente radar instantanée de la cellule, $\varphi_{corps}(t)$ une variable uniformément répartie sur $[0; 2\pi[$, φ_0 un phase à l'origine constante prenant en compte le temps la propagation aller-retour du radar vers la cible et f_D le décalage en fréquence dû à l'effet Doppler défini par :

$$f_D = \frac{2V_{rr}}{\lambda} \quad (2)$$

Le hub, également appelé moyeu, est le système de fixation des pâtes sur l'arbre de transmission du rotor. Cette partie est très signante sur hélicoptère et sur les drones de grandes tailles mais est de conception beaucoup plus simple sur les mini-drones. Cet élément possède une signature caractéristique, avec une décroissante exponentielle de la densité spectrale de puissance autour de la fréquence du corps de l'aéronef. En conséquence, il peut être modélisé comme un nuage de points réfléchissants. Il est composé de M points pour un angle donné du disque comprenant le rotor. La distance radiale d'un point sur ce disque est définie par une fonction de densité suivant une loi exponentielle :

$$f_r(r) = \frac{2}{r_{hub}} e^{-2r/r_{hub}} \quad (3)$$

La répartition angulaire des points suit la fonction de densité f_θ suivante :

$$f_\theta(\theta) = \frac{1}{4}\mathcal{U}\left(\left[\theta_{pale} - \frac{\pi}{N}, \theta_{pale} + \frac{\pi}{N}\right]\right) + \frac{3}{4}\mathcal{N}(\theta_{pale} + \beta, \Delta\theta)$$

où $\mathcal{U}$ représente la loi uniforme, $\mathcal{N}$ la loi normale, θ_{pale} est l'angle de la pale considérée à t=0, β est un décalage angulaire propre à chaque aéronef et $\Delta\theta$ un terme de correction de l'ordre de 1°. Enfin, la répartition verticale des points peut être approximée par une répartition uniforme :

$$f_z(z) = \mathcal{U}([0, h_{hub}]) \quad (4)$$

avec h_{hub} la hauteur du hub. Une fois que les positions initiales des points brillants du hub sont fixées, ils sont mis en rotations à la même vitesse de rotation que les N pales. La réponse totale du hub est alors :

$$s_{hub}(t) = \alpha_{hub} e^{i(\varphi_0+\varphi_r+2\pi f_D t)} \sum_{j=1}^{MN} e^{-2i\overrightarrow{k_r}.\overrightarrow{r_j}} \quad (5)$$

où la réflectivité α_{hub} est ajustée tel que $var(s_{hub}(t)) = \sigma_{hub}$, σ_{hub} étant la SER du hub.

Les pales peuvent être essentiellement vues comme des réflecteurs spéculaires périodiques bien que dans certains cas, elles incluent également une composante continue provenant de l'extrémité de la pale qui peut avoir une réflectivité quasi-isotrope. Elles peuvent être modélisées comme un objet unidimensionnel de longueur L avec une réflectivité constante α par unité de longueur avec une vitesse angulaire Ω_r. Elles sont fixées à une distance r_{hub} de l'axe de rotation. Les forces aérodynamiques et de pesanteur induisent une déformation lors des phases de vol selon l'axe Z, qui peut s'exprimer par $z_b(r, \theta(t))$ comme présenté sur la *Figure 1*.

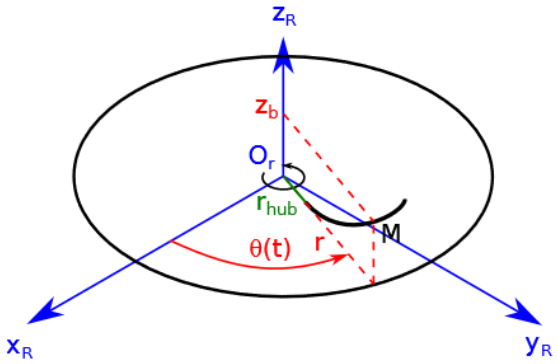


Figure 1 : Schéma présentant la déformation d'une pale selon l'axe Z.

La position du point M, élément unitaire constituent la pale s'exprime alors :

$$\overrightarrow{O_rM}(t) = r\cos(\theta(t))\overrightarrow{e_{xr}} + r\sin(\theta(t))\overrightarrow{e_{yr}} + z_b(r, \theta(t))$$

.

Pour des rotors de petite taille, comme les mini-drones, les pales peuvent être considérées comme une tige droite et le terme $z_b(r, \theta(t))$ peut être négligé [5]. Pour les rotors de grande taille, l'expression de z_b doit être prise en compte, et un phénomène d'oscillation se crée lors des phases de vol en mouvement, ce qui rend l'expression plus complexe [3]. Dans notre cas, nous pouvons donc considérer l'expression du signal d'une pale de la manière suivante :

$$s_{pale}(t) = \alpha_{pale} e^{i(\varphi_0+\varphi_r+2\pi f_D t)} \int_{r_{hub}}^{L} e^{-2i\overrightarrow{k_r}.\overrightarrow{O_rM(t)}}$$

Il faut noter que la valeur de α_{pale} sera différente entre pale en mouvement avancement et une pale en mouvement reculant parce que les bords d'attaque des pales diffèrent forcement de part et d'autre.

Le signal rétrodiffusé par un rotor s'écrit donc :

$$s_{rotor}(t) = s_{hub}(t) + \sum_{i=1}^{N_{rotor}} s_{pale}\left(t + (i-1)\frac{2\pi}{N\Omega_r}\right) + s_{extremite}\left(t + (i-1)\frac{2\pi}{N\Omega_r}\right)$$

Finalement, la réponse totale d'un aéronef à voilure tournante est :

$$s_{drone}(t) = s_{corps}(t) + \sum_{j=1}^{N_{rotor}} r_{rotor,j}(t) \quad (6)$$

Pour la plupart des mini-drones le nombre de rotors, $N_{rotor} = 4$ ou $N_{rotor} = 6$, et sur les hélicoptères $N_{rotor} = 2$.

Lorsque le radar a un échantillonnage de la fréquence Doppler bien plus important que la période de rotation du rotor, les flashs des différentes pales sont clairement visibles sur le spectrogramme. La figure ci-dessous présente des spectrogrammes d'un même hélicoptère obtenus à partir d'un radar aéroporté en bande X en enregistrement réel et en simulation.

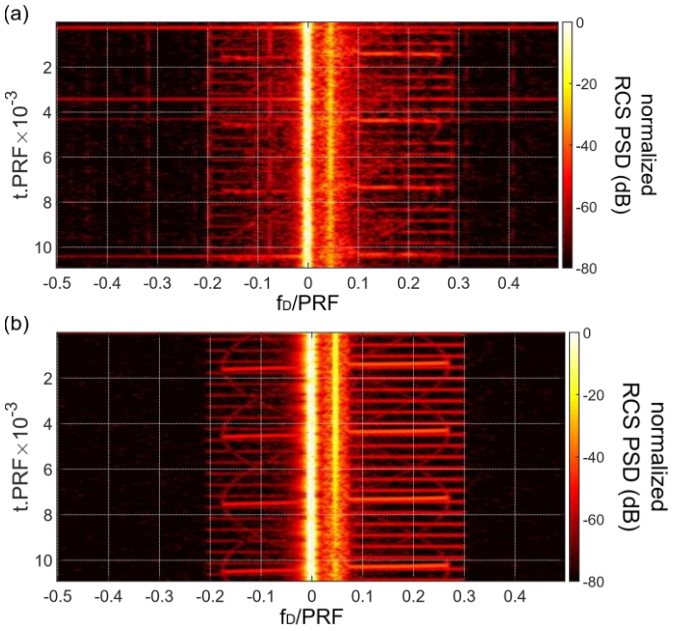


Figure 2 : Spectrogramme d'un hélicoptère obtenu par un radar aéroporté en bande X (en haut) et par simulation (en bas).

Les flashs de pales provenant du rotor principal ainsi que du rotor anti-couple sont très nettement visibles. Les techniques radar classiques de détection de pic dans un environnement de bruit gaussien donnent de bons résultats pour la détection d'hélicoptères [9].

Dans le chapitre suivant, nous verrons que face à un rotor tournant beaucoup plus vite et en utilisant un radar beaucoup moins performant, les spectrogrammes obtenus montrent que les flashs de pales sont beaucoup plus difficiles à discerner, rendant les méthodes évoquées précédemment caduques.

1.3 Enregistrement de signaux radar et analyse du micro-Doppler sur pales de référence

Afin d'obtenir facilement un grand volume de données non protégées sur différents drones à un coût réduit, nous faisons le choix d'utiliser un radar bande K à 24 GHz en mode continu (CW). Bien que n'ayant pas de capacité à mesurer la distance, cette forme d'onde présent l'avantage d'obtenir une signature micro-Doppler avec un échantillonnage rapide, là où une forme d'onde radar FMCW (Frequency Modulated Continuous Wave) rendrait la signature bien plus complexe. En effet, une forme d'onde radar à impulsions nécessiterait une fréquence de répétition des impulsions très importante afin d'obtenir un échantillonnage en fréquence Doppler suffisant, ce qui n'est pas accessible pour un radar bas coût.

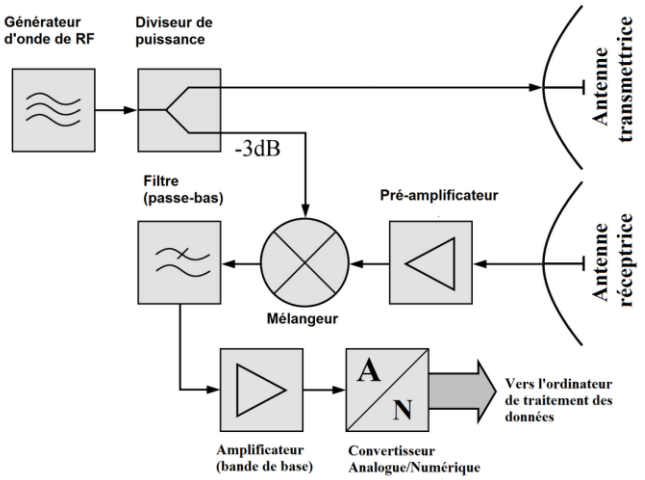


Figure 3 : Diagramme d'un radar FMCW/CW. Source: Charly Whisky, Pierre CB.

Les radars FMCW/CW [10] compacts sont utilisés dans de nombreuses applications : capteurs de mouvement pour l'ouverture automatique des portes, contrôles de vitesse par les forces de l'ordre, assistance à la conduite pour l'automobile. En mode CW, leur fonctionnement est très simple : une fréquence fixe est émise par une antenne d'émission, tandis que le signal reçu par l'antenne de réception est directement démodulé par le signal émis avant d'être numérisés. Tous les signaux renvoyés par des objets fixes par rapport au radar ont donc une fréquence nulle, alors que les objets mobiles renvoient, à cause de l'effet Doppler, des signaux décalés à une fréquence f_D définie dans l'Eq. (1). Notons que plus la longueur d'onde est faible, plus le décalage Doppler sera important pour une vitesse donnée. Cela avantage un radar en bande K par rapport à des radars fonctionnant en bande plus basse comme les radars en bande X. Le traitement du signal consiste simplement à appliquer une transformée de Fourier glissante avec un nombre de points qui doit être

sélectionné en fonction de la nature du signal, avec un recouvrement plus ou moins important, et éventuellement une pondération, afin d'obtenir un spectrogramme temps-fréquence.

Le radar utilisé pour l'acquisition de données est composé d'une carte d'évaluation RFBeam ST200 munie d'un module K-MC1.

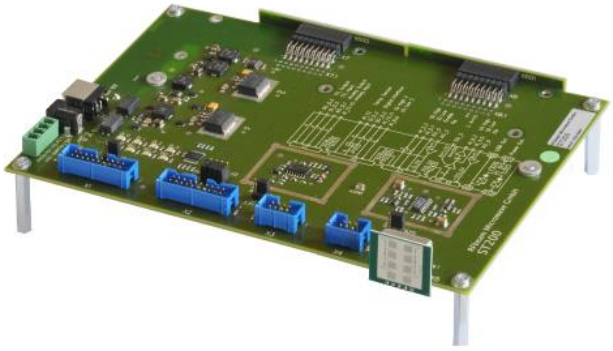


Figure 4 : Carte d'évaluation RFBeam ST200

Les paramètres du radar utilisés pour les acquisitions avec ce module sont les suivants :

- Pe = 16,5 dBm
- Fréquence d'émission = 24,125 GHz
- Ouverture angulaire = 12°x25°
- Fréquence d'échantillonnage = 125 kHz

Avant de faire des enregistrements face à des drones, le radar a d'abord été utilisé pour enregistrer le signal deux hélices dont nous maitrisons la vitesse de rotation. La rotation des deux hélices est effectuée avec un moteur de 5V contrôlé par un puce L293D et un Arduino. Pour cet enregistrement, la vitesse de rotation de l'hélice est de 5000 rpm (83 Hz). A la fréquence d'échantillonnage de 125 kHz, un tour complet prend donc environ 1500 points.

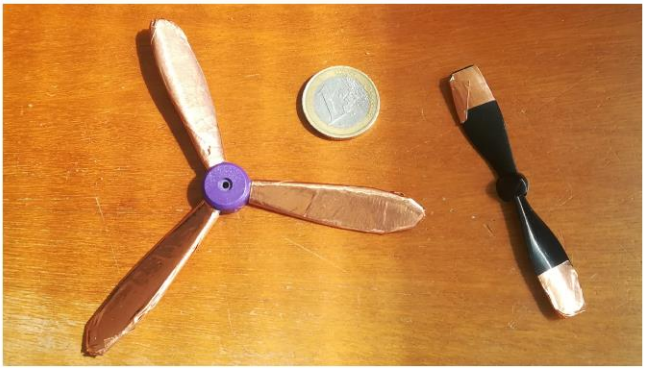


Figure 5 : Hélices à 2 et 3 pales recouvertes de cuivre pour augmenter leur réflectivité

Le nombre de points utilisé pour la transformée de Fourier (DFT) est un paramètre crucial. Pour un signal continu et stable en fréquence, plus le temps d'intégration est important, plus la résolution fréquentielle et le rapport signal sur bruit (RSB) augmente.

Pour un signal non continu ou variable en fréquence, un temps d'intégration trop élevée entrainera une défocalisation du signal qui apparaitra alors comme flou sur le spectrogramme. Pour ces premières mesures, nous prenons 256 points d'intégration.

Les figures suivantes présentent les spectrogrammes obtenus sans couverture cuivre sur la pale, et avec une seuil pale recouverte de cuivre. Nous observons comme prévu une intensité plus importante du flash de la pale cuivrée.

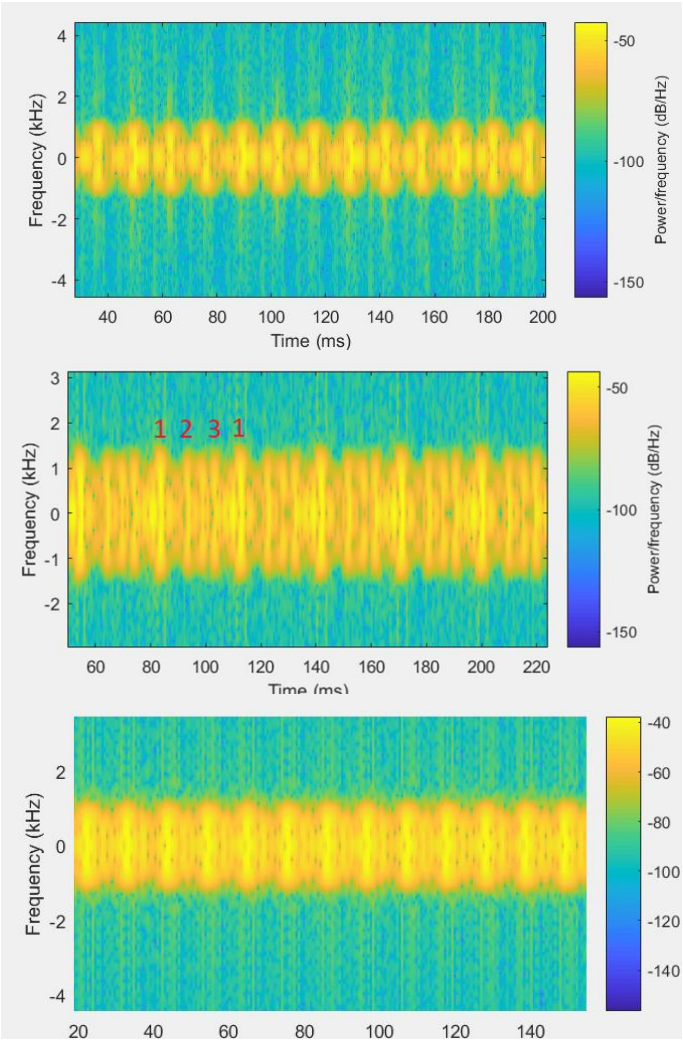


Figure 6 : Spectrogramme obtenu sur hélice sans cuivre (en haut) et sur hélice avec une pale recouverte de cuivre (au milieu) et avec les 3 pales recouvertes de cuivre (en bas)

La Figure 7 suivant présente le spectrogramme obtenu face à l'hélice à deux pales recouvertes de cuivre.

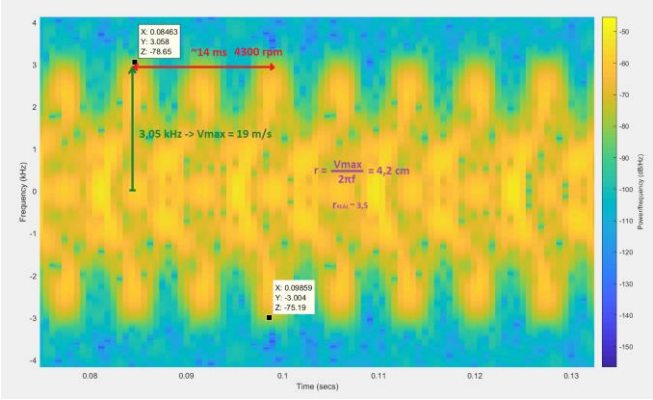


Figure 7 : Spectrogramme d'une hélice à deux pales

D'après l'amplitude du signal dans le spectrogramme, on peut obtenir le décalage Doppler et donc, en appliquant l'équation (1), la vitesse tangentielle maximale des pales, ce qui nous donne une valeur d'environ 19 m/s. Ensuite, on peut obtenir la fréquence de rotation des pales à partir de la fréquence des flashes sur le spectrogramme. On obtient alors une période de rotation d'environ 14 ms, c'est-à-dire 4300 rpm. Avec ces deux grandeurs, on peut estimer la longueur des pales de la façon suivante :

$$r = \frac{V_{tangentielle}}{2\pi f_{angulaire}} = 4{,}2cm$$

La valeur calculée pour la longueur des pales est proche de la valeur réelle, soit environ 4 cm.

1.4 Enregistrement de signaux radar face à des drones et traitement du signal

La campagne de mesures face à des mini-drones a été réalisée en intérieur, dans un gymnase. De ce fait, en l'absence de signal GPS, nous ne disposons de la position des drones lors de l'enregistrement. Les enregistrements ont été effectués face à 7 drones différents : DJI Inspire, DJI Mavic 2, DJI Mavic Mini, DJI Mavic Air, DJI Matrice, Drone FPV, Drone-Act SeeAll.

Pour obtenir les spectrogrammes, nous intégrons le signal sur 512 points. L'horizon temporel définissant la taille des spectrogrammes envoyés à l'algorithme de reconnaissance sera déterminé en fonction des performances de classification. Il existe également deux techniques pour normaliser l'échelle de puissance des spectrogrammes : soit une fenêtre fixe prédéterminée, soit une fenêtre variable prenant le minimum et le maximum de chaque spectrogramme.

Les Figure 8 et Figure 9 présentent les spectrogrammes obtenus pour deux mini-drones avec un horizon temporel de 146 points. Dans la suite, nous avons choisi un horizon temporel de 15 points. Cela permet de segmenter le signal enregistré au cours des vols de façon plus conséquente. C'est un compromis entre le nombre d'images dans la base de données et la quantité d'informations présente sur chacune. Les tenseurs en entrée du réseau de neurones seront donc de dimension (512,15). La normalisation d'échelle en fenêtre variable sera retenue. Finalement, le tableau suivant présente un récapitulatif de la base de données à notre disposition :

Nom du drone	Nombre d'images
DJI Mavic 2	10873
DJI Mavic Air	9267
DJI Inspire	9342
DJI Mini	13508
Drone FPV (Racer)	13492
Drone-Act See-All	9637
DJI Matrice	10247
Bruit thermique	28773

Tableau 1 : Base de données de signaux radar

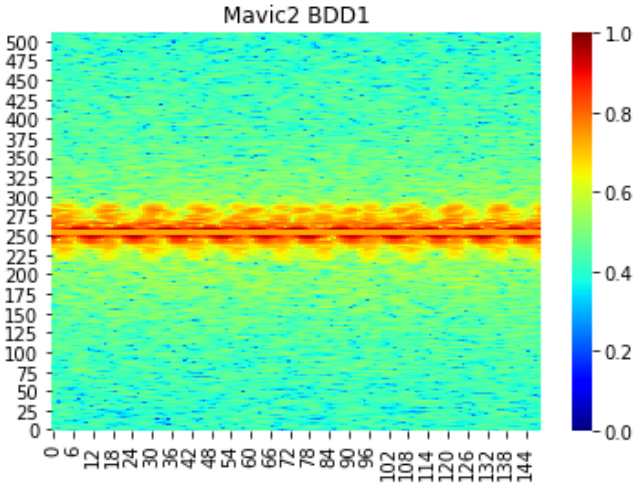


Figure 8 : Spectrogramme d'un DJI Mavic 2

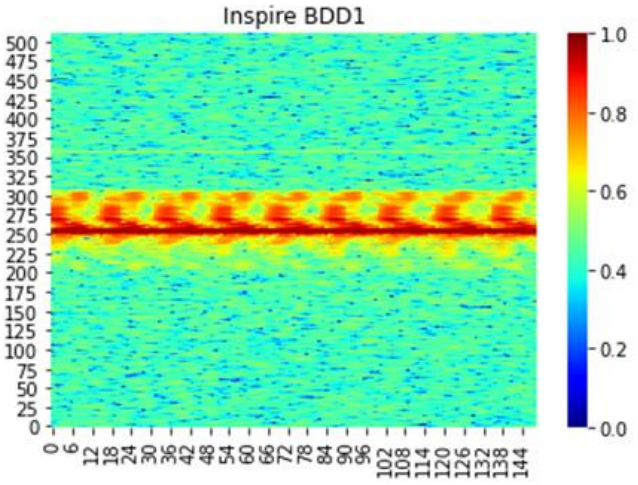


Figure 9 : Spectrogramme d'un DJI Inspire

2 Algorithmes de détection et classification par machine learning

2.1 Méthode proposée

Nous avons ainsi obtenu une base de données d'environ 80000 images temps-fréquence de dimension (512,15). Cette base de données sera aléatoirement séparée en une base d'apprentissage (60%), une base de test (20%) et une base de validation (20%). Notre objectif sera d'étudier dans un premier temps la détection d'un drone, c'est-à-dire le fait de différencier un drone du bruit thermique. Pour ce faire nous utiliserons un réseau de neurones qui sera décrit plus loin. C'est donc une classification binaire. Dans un second temps nous étudierons l'identification du drone (parmi les 7 classes présentées) grâce à ce même réseau de neurones. Cette fois-ci nous nous retrouverons face à un problème de classification multi-classe.

Différents algorithmes proposés dans la littérature montrent que les réseaux de neurones permettent d'obtenir de très bonnes performances en classification sur mini-drones [11][12]. Nous nous concentrons dans cette étude sur un réseau de neurones peu profond de manière à limiter autant que possible la complexité calculatoire du traitement. Il est constitué de différentes couches de convolution et de pooling ainsi que de connexions résiduelles qui ont permis d'améliorer les performances de l'ordre de 5%. Le réseau de neurones convolutif utilisé est illustré sur le *Tableau 2*.

Couche	Type	Taille	Kernel	Filtres
1	Input	512 x 15	-	-
2	Conv2D	512 x 15	3 x 3	32
3	Conv2D	512 x 15	3 x 3	32
4	Add	512 x 15		
5	Max-pool	255 x 7	2 x 2	
6	Conv2D	255 x 7	3 x 3	32
7	Conv2D	255 x 7	3 x 3	32
8	Add	255 x 7		
9	Max-pool	127 x 3	2 x 2	
10	Conv2D	127 x 3	3 x 3	32
11	Conv2D	127 x 3	3 x 3	32
12	Add	127 x 3		
13	Max-pool	64 x 2	2 x 2	
14	Conv2D	64 x 2	3 x 3	32
15	Conv2D	64 x 2	3 x 3	32
16	Add	64 x 2		
17	Max-pool	32 x 1	2 x 2	
18	Flatten	1 x 1024	-	-
19	Dense	1 x 2	-	-
20	Softmax	1 x 2	-	-

Tableau 2 : réseau de neurons utilisé

Le modèle retenu a été entraîné en utilisant une taille de batch de 32 avec l'optimiseur Adam. L'apprentissage s'effectuera sur 20 itérations, en mettant en œuvre l'arrêt conditionné de l'entraînement.

2.2 Entraînement et résultats en détection

Une fois les hyper paramètres de la base de données ainsi que l'architecture du réseau de neurones fixés, nous pouvons réaliser la phase d'entraînement du réseau de neurones. Pour rappel, la détection est une discrimination binaire entre la présence d'un drone et l'absence de drone (bruit thermique seul). Les performances obtenues au cours de l'entrainement sont résumées dans la matrice de confusion et les courbes d'apprentissage présentées sur la Figure 10.

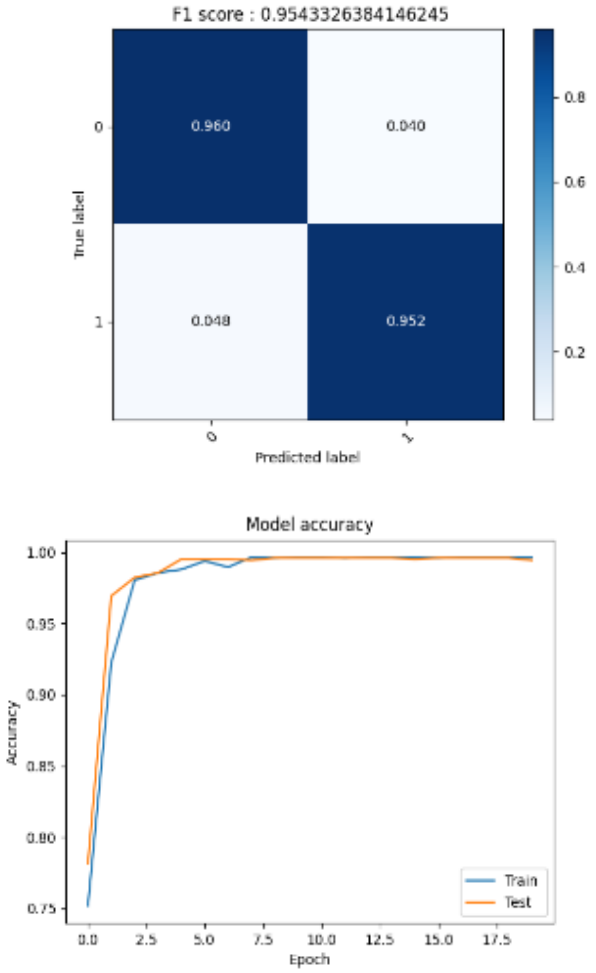


Figure 10 : matrice de confusion pour la détection binaire (en haut) et courbes d'apprentissage (en bas)

Lorsque que le réseau de neurones est correctement entraîné, il est possible de l'utiliser tel quel (les poids sont alors figés) pour réaliser la tâche de détection. Cette partie se rapprochera de ce que l'on retrouve dans un contexte opérationnel puisque l'algorithme traite des passes complètes de vol, en prenant en compte la dimension temporelle. Dans nos essais, les drones effectuent des aller-retours sur une distance d'une trentaine de mètre, mais nous n'avons pas de mesure précise de la distance, du fait de l'utilisation du mode CW. Il nous est cependant possible d'analyser l'allure de la moyenne locale des scores de détection au cours de

l'essai. Cette moyenne correspond à la moyenne du score en sortie du réseau de neurones sur une série d'images temps-fréquence consécutives.

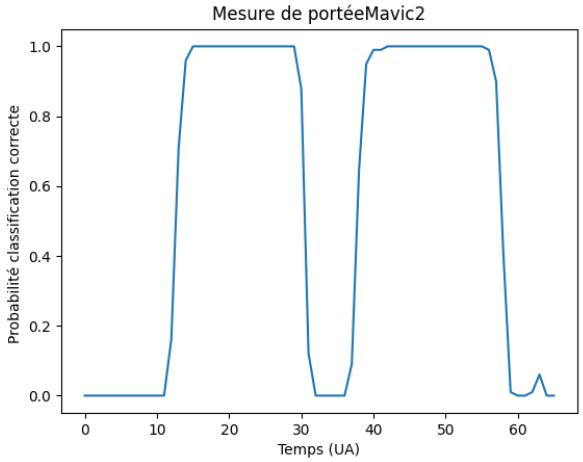


Figure 11 : probabilité de détection du drone DJI Mavic 2 au cours du temps

En se concentrant sur le vol du Mavic 2 on peut observer que l'allure de la courbe de score du drone est tout à fait en accord avec les mouvements d'aller-retours qui ont été réalisés au cours des essais. On peut conclure que l'algorithme de classification se comporte comme attendu (aucune classification au-delà d'une certaine distance). Pour caractériser cette valeur de portée avec précision, il nous manque cependant la distance à l'instant T ≃ 30UA. Nous pouvons cependant estimer qu'elle se situe à une vingtaine de mètre pour ce drone.

2.3 Entraînement et résultats en classification

De la même façon, en modifiant cette fois la couche de sortie du réseau de neurones convolutif, voici les performances obtenues en identification au cours de l'entraînement.

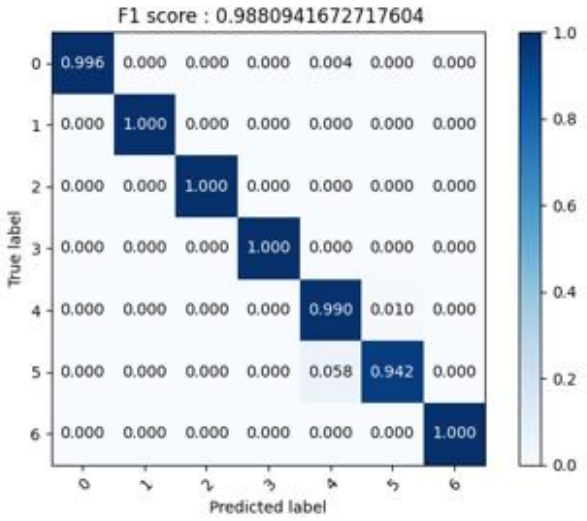


Figure 12 : matrice de confusion pour l'identification des drones

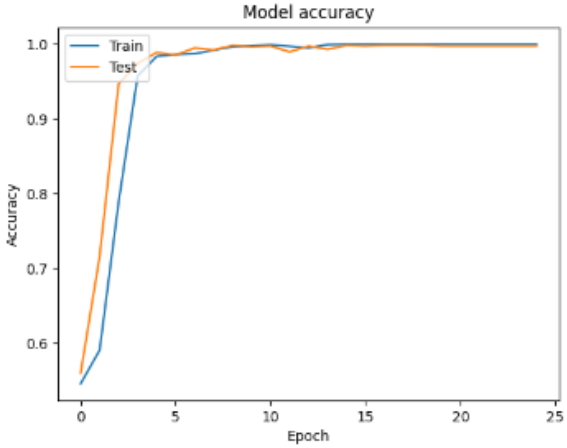


Figure 13 : courbes d'apprentissage pour le réseau de detection d'identification des drones

En utilisant la moyenne locale des scores de classification, nous chercherons à vérifier la robustesse de la classification à la distance.

2.4 Capacité de détection de nouveauté

Il existe de nombreux types de drones, qui sont parfois modifiés pour les besoins adverses, il sera sans doute difficile, voire impossible d'avoir une base de données exhaustive contenant tous les drones que l'algorithme sera amené à rencontrer. Il est donc important que la détection d'un drone soit robuste à un drone inconnu, c'est-à-dire un drone qui n'est pas présent dans la base de données d'apprentissage.
Pour cela, nous allons retirer toutes les données de l'un des drones de la base de données (le DJI Mavic 2 dans l'exemple qui suit). Nous allons donc reprendre la phase d'entraînement avec les données bruit thermique ainsi que celles de tous les autres drones sauf celles du drone étudié. Une fois l'entraînement terminé, nous allons utiliser le réseau de neurones pour tenter de détecter la présence d'un drone au cours du vol du drone DJI Mavic 2 qui est donc inconnu du réseau. Nous comparerons ensuite ces données avec le même vol lorsque le drone DJI Mavic 2 se situe bien dans la base de données d'apprentissage pour bénéficier d'une base initiale et quantifier l'intérêt d'avoir la base de données la plus complète possible. La Figure 14 présente les courbes de probabilité de détection avec et sans le drone Mavic 2 dans la base d'apprentissage.

Les deux courbes ont une allure similaire et nous remarquons que, logiquement, l'ajout du drone à la base de données d'apprentissage permet d'améliorer légèrement les performances. Avoir le drone cible dans la base de données d'apprentissage améliore donc sa détectabilité, mais son absence ne remet pas en cause l'utilisation de cet algorithme à des fins de détection.

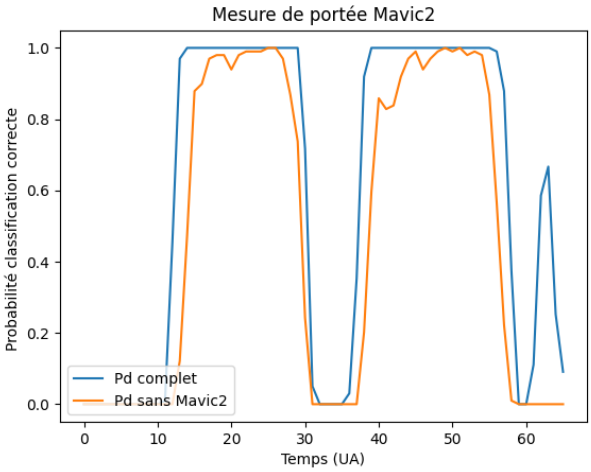


Figure 14 : probabilité de détection face à un Mavic 2 avec et sans Mavic 2 dans la base d'apprentissage

2.5 Implémentation en temps réel

Les temps d'inférence de ces deux réseaux de neurones ont été mesurés sur différents GPUs NVIVIDIA. Le but est d'estimer la ressource de calcul nécessaire à l'utilisation de ces algorithmes. Il est en effet important de traiter les signaux radar reçus en temps réel afin d'identifier une menace et d'y répondre dans un délai le plus court possible.

Ces mesures ont été faites en utilisant uniquement un appel de prédiction dans le framework Tensoflow. Ces temps sont donc significativement supérieurs à ce que que serait une implémentation plus optimal utilisant par exemple TensorRT. Trois processeurs graphiques ont été évalués : carte RTX GeForce 2080 Ti, un module SoC NVIDIA Jetson Nano et sur un module SoC NVIDIA Jetson AGX Orin. Les temps d'inférence moyens sont les suivants :

Matériel	Temps d'inférence	TDP max
RTX GeForce 2080 Ti	13 ms	250W
Jetson Nano	860 ms	10W
Jetson Orin	70 ms	65W

Ces temps ne comprennent que l'inférence en elle-même. Ils ne sont dont pas négligeables par rapport à l'acquisition du signal qui est de 140 ms. Nous voyons néanmoins qu'avec une implémentation non optimisée, un module AGX Orin, qui est très adapté pour les systèmes embarqués compacts, permet de réaliser l'inférence en un temps deux fois plus court que l'acquisition des données.

3 Conclusion

Dans ce papier, nous avons montré que les signaux rétrodiffusés par un aéronef muni de rotors sont complexes. Sur des données enregistrées à l'aide d'un radar compact à faible coût, il est néanmoins possible d'obtenir des performances intéressantes en détection et classification à l'aide d'un algorithme à base d'un réseau de neurones convolutif muni de connexions résiduelles. Une caractérisation plus fine des performances en fonction du rapport signal sur bruit reste toutefois à réaliser, mais ces premiers résultats de la combinaison entre un prétraitement temps-fréquence et de cette architecture de réseau de neurones permettent d'obtenir de bons résultats en détection et en identification au moyen d'une capacité de calcul limitée. Nous avons également montré que ce type d'approche est relativement robuste à l'absence du drone cible dans la base d'apprentissage.
Les travaux futurs porteront d'une part sur l'acquisition de signaux à partir de radars plus performants en extérieur, et d'autre part à l'étude du gain en temps d'inférence obtenu à l'aide d'une implémentation plus optimisée.

4 Bibliographie

Références

[1] Liu, H., Wei, Z., Chen, Y., Pan, J., Lin, L., & Ren, Y. (2017, April). Drone detection based on an audio-assisted camera array. In 2017 IEEE Third International Conference on Multimedia Big Data (BigMM) (pp. 402-406). IEEE.

[2] Mezei, J., Fiaska, V., & Molnár, A. (2015, November). Drone sound detection. In 2015 16th IEEE International Symposium on Computational Intelligence and Informatics (CINTI) (pp. 333-338). IEEE.

[3] Á. D. de Quevedo, F. I. Urzaiz, J. G. Menoyo and A. A. López, "Drone Detection With X-Band Ubiquitous Radar," *2018 19th International Radar Symposium (IRS)*, Bonn, Germany, 2018, pp. 1-10, doi: 10.23919/IRS.2018.8447942.

[4] Ochodnický, J., Matousek, Z., Babjak, M., & Kurty, J. (2017, May). Drone detection by Ku-band battlefield radar. In 2017 International Conference on Military Technologies (ICMT) (pp. 613-616). IEEE.

[5] Wehling, J. H. (2005). Multifunction millimeter-wave systems for armored vehicle application. IEEE transactions on microwave theory and techniques, 53(3), 1021-1025.

[6] Hudec, P., Raboch, J., Randus, M., Hoffmann, K., Holub, A., Svanda, M., & Polivka, M. (2009,

September). Microwave radar sensors for active defense systems. In 2009 European Radar Conference (EuRAD) (pp. 581-584). IEEE.

[7] Wey, P., Fleck, V., & Chanteret, P. Y. (2001, May). Analysis of active protection systems: when athena meets arena. In Proceedings of the 19th International Symposium on Ballistics.

[8] Point, G., Degurse, J. F., Savy, L., Montécot, M., & Milin, J. L. (2021). Modelling the radar signature of rotorcraft. IET Radar, Sonar & Navigation, 15(8), 867-88.

[9] Misiurewicz, J., Kulpa, K., & Czekala, Z. (1997). Analysis of recorded helicopter echo.

[10] Stove, A. G. (1992, October). Linear FMCW radar techniques. In IEE Proceedings F (Radar and Signal Processing) (Vol. 139, No. 5, pp. 343-350). IET Digital Library.

[11] Brooks, D. A., Schwander, O., Barbaresco, F., Schneider, J. Y., & Cord, M. (2018, June). Temporal deep learning for drone micro-Doppler classification. In *2018 19th International Radar Symposium (IRS)* (pp. 1-10). IEEE.

[12] Park, D., Lee, S., Park, S., & Kwak, N. (2021). Radar-spectrogram-based UAV classification using convolutional neural networks. *Sensors*, *21*(1), 210

Loss Function Design For Training Robust Radar Detectors Using Deep Learning

Noé Lallouet
LAMSADE, Université Paris-Dauphine
Thales DMS
Paris, France
noe.lallouet@thalesgroup.com

Tristan Cazenave
LAMSADE, Université Paris-Dauphine
Paris, France
tristan.cazenave@lamsade.dauphine.fr

Cyrille Enderli
Thales DMS
Elancourt, France
cyrille-jean.enderli@fr.thalesgroup.com

Stéphanie Gourdin
Thales DMS
Elancourt, France
stephanie.gourdin@fr.thalesgroup.com

***Abstract*—Recent advances in research show interesting potential in using deep neural networks to perform the task of radar target detection. For radar applications, especially in the military domain, the detection method is designed to follow the Neyman-Pearson criterion with the aim to maximise the detection probability while keeping the false detection rate controlled. While standard CFAR (Constant False Alarm Rate) detection is designed to fit this need, it is not the case of neural networks that do not naturally prioritize detection over false alarm rate. In this paper, we propose an overview and a comparison of different loss functions, namely Tversky Loss and a loss based on constrained optimization, for training deep CNNs on the problem of radar target detection, with the objective to get a better compromise between false alarm rate and detection. We then demonstrate that the models obtained with these methods outperform the baseline CNN model as well as classic CFAR detectors. The developed models are compared on the detection probability (P_D) and false alarm probability (P_{FA}) criteria on exoclutter[1] environments. Model evaluation on thermal noise is an important step for validating a detector and is rarely explored in related research. It is found that training the neural network with a loss function constrained by the expected false alarm probability provides higher detection probability at a fixed P_{FA} than the baseline models. The advantages and shortcomings of training the detector with the Tversky loss function are also highlighted.**

***Index Terms*—Radar, target detection, deep learning, optimization**

I. Introduction

Target detection is one of the most fundamental problems in radar signal processing. On exoclutter environments, target detection consists in solving the following binary decision problem :

$$\begin{cases} H_0 : y(t) = \nu(t) : \text{absence of target} \\ H_1 : y(t) = x(t) + \nu(t) : \text{presence of target} \end{cases} \quad (1)$$

where $y(t)$ is the received signal, $\nu(t)$ is the thermal noise signal and $x(t)$ is the signal of the target. This paper addresses detection in the setting of a Pulse-Doppler radar. The problem of target detection is optimally solved with statistical signal processing methods such as CFAR [1], but these methods require strong assumptions about environment and target unicity that, in real situations, are not always observed.

The use of deep learning applied to radar target detection has risen in recent years, having shown promising results. However, a critical aspect that remains underexplored is the ability to build reliable detectors and provide performance guarantees. Our work contributes to filling this gap in research by investigating ways to train the model with the goal of satisfying a constraint on P_{FA}. Moreover, scarce are the research works solely focusing on deep learning-based detector performance evaluation over exoclutter environments. This crucial assessment needs to be performed in order to validate neural detectors operationally.

It is found that radar detection performance improves when training the neural network with different loss functions than the ones that have been focused on in related research.

II. Related Works

In recent years, the ever-increasing available computation power and existence of large datasets have enabled deep learning models to revolutionize the way numerous problems are approached, including natural language processing, time series forecasting, computer vision, etc. In particular, convolutional neural networks (CNNs) have been successfully used to perform different vision tasks, such as image classification ([2], [3], [4]) or segmentation ([5], [6], [7]).

The problem of radar target detection on range-Doppler maps, in the operational setting used in this paper, can be expressed as a single-point image segmentation problem. Indeed, after signal processing, the target appears as a singular point in a 2D array. As such, the use of classic image segmentation architectures applied to radar signal processing is of particular interest. After neural network inference, the post-processing chain performs non-max suppression in order to prevent

[1]Exoclutter refers to a situation where there is an absence of unwanted echoes.

eventual small inaccuracies of around one pixel in the predicted segmentation map.

Today, radar target detection is performed using Constant False Alarm Rate (CFAR) detectors [1]. These detectors estimate the variance of thermal noise in a neighbourhood of the cell under test (CUT) and announce a detection if the value of the CUT is superior to a certain threshold, which is normalized with respect to the estimation of local noise power. The threshold is computed with respect to the estimated noise variance. CFAR detectors, under strong assumptions, are statistically optimal. However, the assumptions in which CFAR detectors are optimal (absence of clutter[2] and/or side lobes, unicity of target, range resolution superior to target size...) are seldom met, resulting in suboptimal target detection performance. Even tough techniques (GO-CFAR, SO-CFAR...) [8] have been developed to mitigate this, there exists a margin to develop detectors more polyvalent than CFAR-based detectors.

Radar signal processing has also been subject to deep learning-based innovation. Deep learning has been successfully used to perform radar waveform recognition [9], automatic target recognition [10], among others. The works of [11], [12] leverage convolutional neural networks (CNNs) to detect targets in the 4D space (range, Doppler, azimuth and elevation), while [13] and [14] focus on CNN detectors on range-Doppler maps. The works of [15] implement a Faster R-CNN network ([16]) in order to automatically detect marine targets.

The development of neural networks capable of guaranteeing a fixed P_{FA} has also been the object of research. Research by [17] promotes a CFAR neural detector by training a network to minimize a statistical distance between the network output of two examples belonging in the H_0 hypothesis (Equation 1). The work of [18] leverages a CNN that detects and masks a target from a range-Doppler map in order to improve the noise estimation performed by a classic CFAR detector. Finally, [19] introduces a novel constrained loss function related to the Neyman-Pearson criterion for marine target detection. The network of [19] outperforms CFAR detector and a CNN trained using the cross-entropy loss. We propose to extend this comparison of a constrained loss against other loss functions. Furthermore, even though we aim to develop a polyvalent detector, the scope of our analysis is restrained to the exoclutter scenario, where comparing detection and false alarm performances is straightforward.

III. Methodology

A simulation model that is able to generate realistic radar echoes is used to create the data necessary for training the model. The radar signal generation tool is validated and generates data which is representative of real data. The benefits of using such a generator are threefold : first, our ability to generate data is only limited by computation and time constraints. Second, it can be chosen to produce data that represents the distribution of real-world applications. Finally, the correctness of the ground truth labels is guaranteed.

The data consists in 100000 range-Doppler maps on which have been added 5 simulated targets. The number of 5 is chosen with the aim of increasing the number of positive examples in the training dataset, thus reducing data imbalance. Varying the number of targets on a training example, by, for example, randomly choosing a number of targets during simulation, and shifting the variance of thermal noise might be of interest in future work. The range-Doppler maps are generated according to a number of scenarii, including both endoclutter and exoclutter situations. The motivation for training the model on various noise and clutter profiles resides in the will to create a model able to generalize detection on exoclutter environments to more complex ones. Each target has a distance and velocity that is randomly drawn inside the operational domain according to a bivariate uniform law. Additionally, the SNR (signal-to-noise ratio) of the generated images is tuned by varying the radar cross section (RCS) of the targets, thus effectively varying the reflected power received by the radar.

Research works such as [10] and [11] have shown that neural networks, and especially CNNs, are adapted to the task of radar target detection. In accordance with these findings, a CNN architecture is used to perform the task. As the problem of target detection on range-Doppler maps can be formulated as an image segmentation problem, a classic U-Net encoder-decoder structure [6] is leveraged. The U-Net architecture consists in several encoding stages that express the input in a lower-dimensional latent space, which are followed by decoding stages that take the input back to its original resolution. Any two pair of encoding and decoding stages of the same rank k are connected by a *shortcut connection*, that brings back the necessary spatial information. The shortcut connection is an `Add` operator taking as inputs the features map for rank k of the encoder and decoder. The range-Doppler maps are resized to the classical tensor dimensions of 256×256 and given as input to the model. The architecture of the network is detailed in Figure 1. The sigmoid activation function is applied to the final layer activation values, outputting a value between 0 and 1 representing a confidence value of classifying the pixel as background noise or as a target. The neural network has a total of $2,121,481$ parameters.

The network is trained on the aforementioned simulated range-Doppler maps. A training / validation split of 80/20% is used. The test data is generated on-the-fly, and therefore has not been used during training. It consists in a singular target in a simulated exoclutter environment. Neural network testing is performed with 20,000 generated images. The performance metrics such as detection probability P_D and false alarm probability P_{FA} are computed after the detection post-processing chain used in the radar detection pipeline. As such, these metrics are representative of detection performance in real scenarii.

Network training is performed on a Nvidia GTX 1080 Ti using the Adam optimizer and a learning rate decreasing from 7×10^{-3} to 5×10^{-5} for 150 epochs. Model values with

[2]Clutter is defined as unwanted echos received by the radar, e.g. coming from the ground. In our operational setting, ground clutter is expected to cover roughly half of the range-Doppler map.

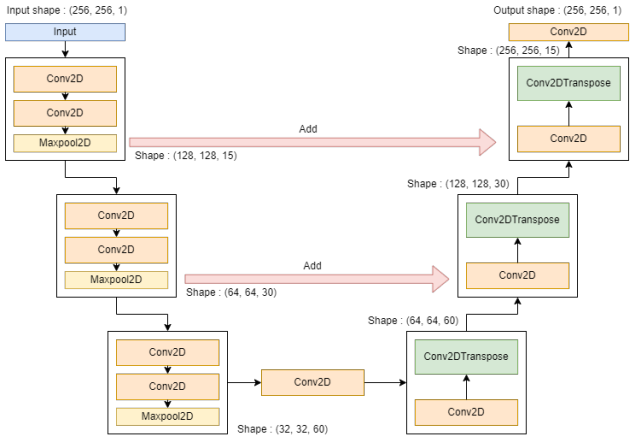


Fig. 1. Architecture of U-Net

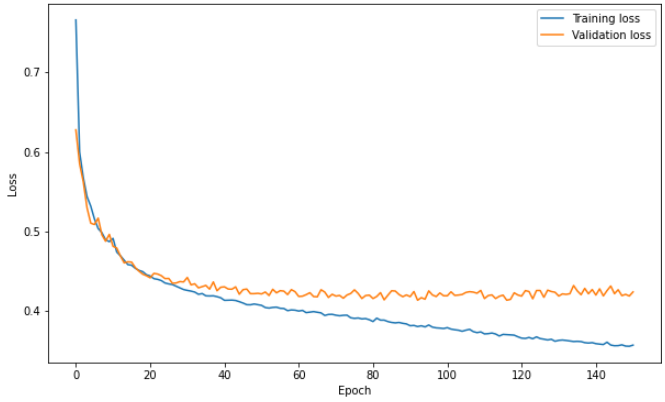


Fig. 2. Evolution of training and validation loss for the baseline CNN model

the lowest validation set loss are selected, in order to reduce overfitting. The training and validation loss evolution curves for the baseline CNN model trained with the Dice loss function is shown in Figure 2. We have not encountered any specific difficulties for model training, as weight convergence remains stable for any training run.

Loss function

A large number of research works applying CNNs to the task of radar target detection use popular loss functions such as cross-entropy, weighted cross-entropy or the Dice coefficient for training. Even though these choices are valid and lead to good detection performances, they are not designed to fit the very specific radar detection problem. As such, it remains to be seen whether training the network with a different loss function can improve P_D at a fixed P_{FA} over baseline models. It has been previously shown [20] that using the Dice coefficient (2) as a loss function during model training leads to improved segmentation accuracy over weighted cross-entropy loss in the case of imbalanced data. Due to the sparse nature of the data, where the positive-to-negative label ratio is 7.6×10^{-5}, it has been chosen to use Dice coefficient as a baseline.

$$D(\hat{y}, y) = \frac{2\sum_i^N \hat{y}_i y_i}{\sum_i^N \hat{y}_i^2 + \sum_i^N y_i^2} \tag{2}$$

where $\hat{y}_i$ and y_i are respectively predicted pixels and ground truth pixels. N is the total number of pixels in an image. $\hat{y}_i$ belongs to the interval $]0; 1[$, while y_i belongs to the set $\{0, 1\}$. When using mini-batch stochastic gradient descent, which is the case of this research work, the value D is calculated as the average of the Dice coefficient for every image in a batch of size m. The CNN trained using the Dice Loss $DL = 1 - D$ as a cost function will be referred to further as **NN-Dice**.

We formulate the hypothesis that convolutional neural networks trained using the Dice coefficient as a loss function show limits in performance due to the fact that the Dice coefficient aims to maximize the intersection over union of ground truth and prediction pixels. As such, the loss function assigns the same weight to false negatives (missed detections) and false positives (false alarms). As the results presented in Section IV show, a network trained using the Dice coefficient results in a high-precision and low-recall detector. It is then of interest to train a detector that exhibits a higher recall, while staying under the required maximum number of false alarms.

We introduce the use of the Tversky Loss [21] during training of the model. Tversky Loss is an extension of Dice loss with added coefficients which enable to control the importance that is given to false positives and false negatives during training. Tverksy Loss has been investigated in automotive radar applications in [22]. Using previous notation, the Tversky Index TI can be written as follows :

$$TI = \frac{\sum_i^N \hat{y}_i y_i}{\sum_i^N \hat{y}_i y_i + \alpha \sum_i^N \hat{y}_i (1 - y_i) + (1 - \alpha) \sum_i^N (1 - \hat{y}_i) y_i}$$

The cost function that is optimized during training is $TL(\hat{y}, y) = 1 - TI(\hat{y}, y)$. It may be noticed that $\sum_i^N \hat{y}_i y_i$ represents true positives (TP), $\sum_i^N \hat{y}_i (1 - y_i)$ false positives (FP) and $\sum_i^N (1 - \hat{y}_i) y_i$ false negatives (FN). Thus, the parameter α may be used to give more or less importance to one or the other during training.

The hyperparameter α is found using a grid search and selecting the model that maximizes P_D at a fixed P_{FA}. To the best of the authors' knowledge, Tversky Loss has not been used during the training of neural networks in the context of radar target detection.

The neural network trained using TL will be referred to as **NN-Tversky**. As it shall be demonstrated in the results section, using Tversky Loss improves the detection probability of the algorithm. However, even though the false alarm rate remains inferior to the value at which the CFAR detector is calibrated, it remains unclear how to guarantee an expected false alarm rate for neural detectors.

False alarm control

Radar detection systems operate under the constraint of producing a fixed number of false alarms during a period of time. The classical value is set at 1 false alarm (FA) per

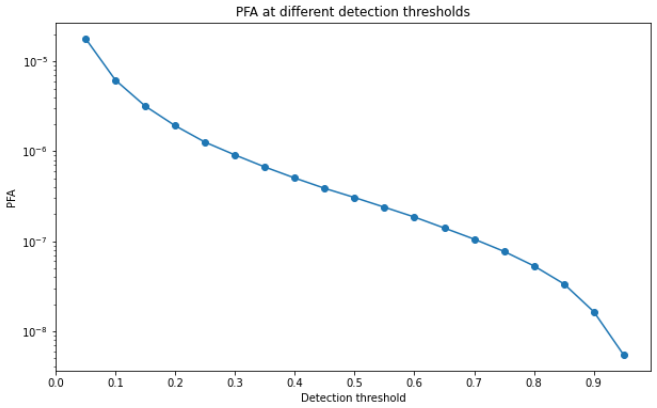


Fig. 3. P_{FA} at varying detection thresholds for the Dice loss function

minute. On exoclutter backgrounds, the dynamic threshold computation guarantees —notwithstanding CFAR loss—a statistically maximal P_D at a fixed P_{FA}. Thus, CFAR detectors are, in that sense, reliable. However, deep learning detectors do not classically take into account the constraint on P_{FA}. Neural detectors trained using Dice Loss or Tversky Loss can be calibrated with respect to P_{FA} by applying a threshold filter over the output layer activations : indeed, the network's output layer activations represent the classification of each pixel as a confidence value between 0 (background noise) and 1 (target). Applying a threshold transforms these values into a binary prediction map. In order to determine this threshold, the neural network is then evaluated on a test set S_1 containing varying thermal noise profiles but no target. The pre-threshold confidence indexes of the network are stored and P_{FA} is computed applying a sliding threshold to these values. The threshold is finally selected according to the desired P_{FA}. A representation of this false alarm threshold tuning for the network trained using the baseline Dice loss function is presented in Figure 3.

The research work of [19] shows that it is possible to develop a loss function based on the Neyman-Pearson criterion in order to train a neural network with respect to a fixed constraint, in our case the false alarm probability. The goal of this constrained optimization is to penalize the neural network for producing an unwanted number of false alarms. As such, the constrained detection problem (P) may be formulated as the following :

$$\begin{aligned} \text{minimize}_w \, \mathcal{L}(\hat{y}, y) \; &= TL(\hat{y}, y) \\ \text{s.t.} \;\; FP(\hat{y}, y) &\leq s \end{aligned} \tag{P}$$

where w are the neural network weights, $TL(\hat{y}, y)$ is the Tversky Loss TL for prediction $\hat{y}$ and ground truth y, and $FP(\hat{y}, y) = \sum_i^N \hat{y}_i(1 - y_i)$. The inequality constraint can be reformulated as an equality one:

$$\begin{aligned} \text{minimize}_w \, \mathcal{L}(\hat{y}, y) \; &= \; TL(\hat{y}, y) \\ \text{s.t.} \; g(\hat{y}, y) = (\frac{1}{s}(FP(\hat{y}, y) - s))^2 &= 0 \end{aligned}$$

This cost function differs from the one used in [19] in the fact that the network is also penalized for producing a number of false alarms that is largely inferior to the threshold. Indeed, setting the penalty to 0 for $FP(\hat{y}, y) < s$ may result in convergence to local minima where both P_{FA} and P_D are close to 0. Particularly, when s is very low, the network may struggle to address the constraint and fall into those local minima. Furthermore, incentivizing the model to produce a number of false alarms closer to the threshold incorporates the idea of threshold calibration that is performed on classical CFAR methods.

The constraint $g(\hat{y}, y)$ may now be incorporated alongside the objective function, in order to go from a constrained problem to an unconstrained one. As such, the updated objective function $\mathcal{L}_P$, using Tversky Loss as a proxy for detection probability, is :

$$\mathcal{L}_P(\hat{y}, y, \lambda) = TL + \lambda g(\hat{y}, y) \tag{3}$$

where λ is an estimate of the Lagrange multiplicator. The neural network trained using Equation 3 will be further referred to as **NN-Lagrange**.

IV. Results

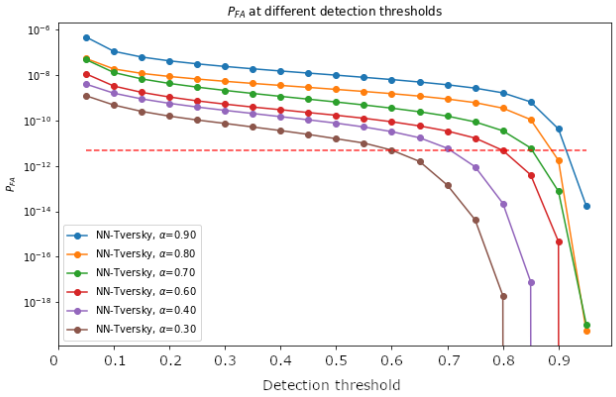


Fig. 4. P_{FA} at various thresholds for NN-Tversky

Figure 4 shows the false alarm probability P_{FA} for different threshold values, for CNN detectors trained using the Tversky loss function with $\alpha \in \{0.1, 0.2, ..., 0.9\}$. It may be observed that, for high values of α, the P_{FA} stays high and only decreases at a large threshold value. The differences between P_{FA} curves for various α is of interest in the setting of radar target detection, as it shall be expanded upon in Section V. For the scenario of this research work, with a singular target on an exoclutter environment, it is decided to choose NN-Tversky with the following hyperparameters : $\alpha = 0.6$ and threshold $= 0.8$.

Figure 5 displays P_{FA} curves against threshold value for the detectors that have been chosen for comparison : NN-Dice, NN-Tversky and NN-Lagrange. P_{FA} for thresholds 0 and 1 are not computed, since values are identical for all models and would hamper figure readability. One may notice that NN-Lagrange exhibits a larger P_{FA} variation across the threshold values in the computed range ($[0.05; 0.95]$) than NN-Dice and

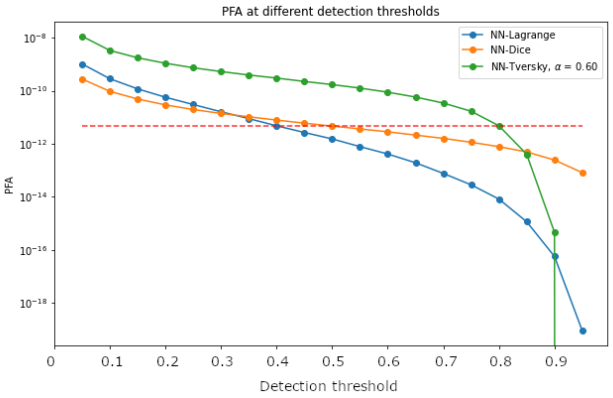


Fig. 5. P_{FA} at various thresholds

NN-Tversky - that is, NN-Lagrange can produce a higher range of P_{FA} values by threshold setting than the other loss functions which seem limited to a narrower range. The final layer activation threshold values are chosen according to the expected number of false alarms.

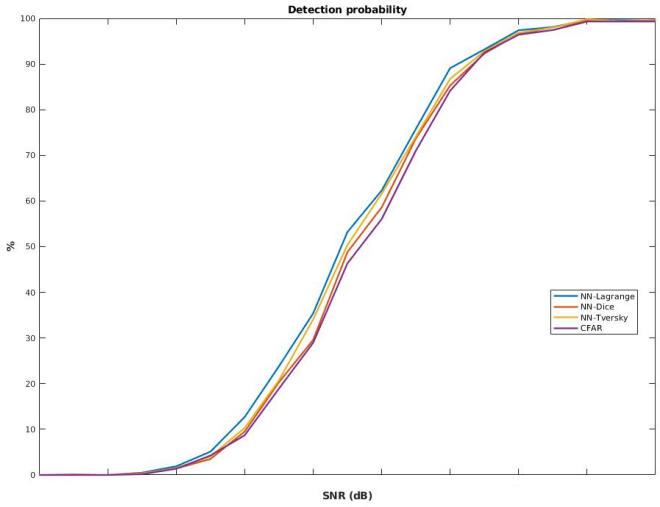


Fig. 6. Detection probability as a function of SNR

Model	CFAR	NN-Dice	NN-Tversky	NN-Lagrange
$P_{FA}(\times 10^{-7})$	1.64	0.951	1.15	0.573

TABLE I
P_{FA} VALUES FOR DIFFERENT DETECTORS

It can be seen, from Figure 6, that NN-Lagrange provides the best detection performances across all methods. All neural detectors improve P_D over the CFAR detector, which detection probability is decreased by the CFAR loss resulting from a potentially imprecise noise variance estimation due to the limited number of noise samples. NN-Lagrange noticeably outperforms NN-Tversky and NN-Dice. Indeed, while the

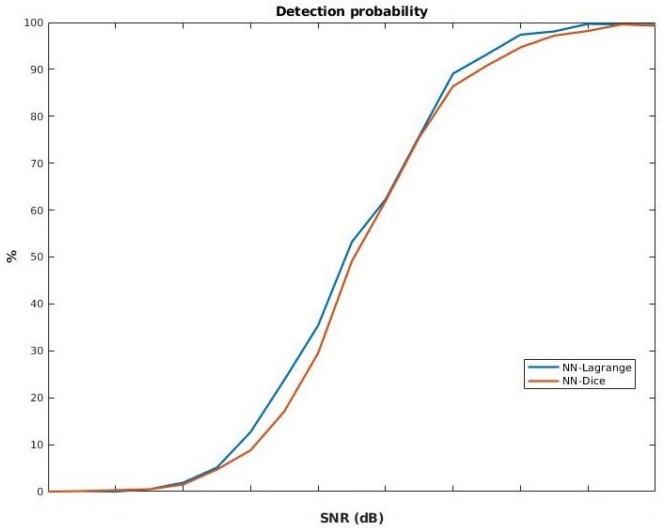


Fig. 7. Comparison of the inclusion of TL vs DL in the constrained objective function

latter two have seen their final layer activation threshold fine-tuned using detection on targetless thermal noise, NN-Lagrange incorporates the constraint on the number of false alarms during training. It is then straightforward to note that there is less information loss with NN-Lagrange. It may also be seen that NN-Tversky boasts a slightly higher detection performance than NN-Dice. Again, this is due to the fact that the network has been trained to produce a higher number of detections, which can be leveraged by fine-tuning the detection threshold.

Table I shows the false alarm probability P_{FA} for the compared models. It can be appreciated that all neural network detectors exhibit a lower P_{FA} than CFAR. P_{FA} values gravitate around the same operating point, with the exception of NN-Lagrange, which remarkably achieves both the highest P_D and the lowest P_{FA}. The relative improvements of the studied loss functions over CFAR can thus be validated.

Ablation study

An ablation study is performed in order to understand the choice of the Tversky Loss TL instead of the Dice Loss DL in the objective function for the constrained optimization problem (P). Two networks are compared : NN-Lagrange, which has been introduced earlier, and a neural network trained by replacing the Tversky Loss TL by the Dice Loss DL :

$$\mathcal{L}_P(\hat{y}, y, \lambda) = DL + \lambda g(\hat{y}, y) \tag{4}$$

Figure 7 shows that, for an identical P_{FA}, the network trained using TL in the objective function displays a higher P_D than the detector trained with DL. This difference in P_D is explained by the fact that the former network proposes more detection candidates due to the nature of TL when α is close to 1. This higher number of detection candidates has no influence on P_{FA} due to the constraint.

V. Discussion

As it has been discussed in Section IV, NN-Tversky shows, at a fixed P_{FA}, higher detection performance than NN-Dice. This superiority may be explained by the fact that the parameter α Tversky Loss incorporates provides an additional degree of liberty during the network calibration process. One is able to select both the optimal α and the optimal final layer activation threshold in order to maximize detection probability at a fixed P_{FA}. This versatility can also be used to calibrate the network on other more difficult environments, containing perturbations and/or interactions. However, selecting an optimal value of α requires network retraining, while selecting an optimal threshold is performed after training. Whether model training for a specific α may be performed in little time, using, e.g., transfer learning, will be the subject of further investigations.

However, the hyperparameters yielding optimal performance for NN-Tversky are empirically found and tuned and are not backed by theoretical guarantees. NN-Lagrange addresses this shortcoming by bringing a notion of stability, as the network is trained to respect a given P_{FA}. However, we show that the constraint, while satisfied during training, is not guaranteed to be at inference, because the loss function is not computed. Giving theoretical P_{FA} guarantees for CNN detectors at inference time is still the object of research.

It is interesting to notice that, even though the network has been trained on a dataset containing thermal noise as well as ground clutter, in hopes of having a model capable of generalizing to complex environments, the detection performances do not decrease compared to a network that has been trained on thermal noise alone. This can be explained by the fact that adding ground clutter or other perturbations brings robustness to the model.

VI. Conclusion

In this paper, a deep learning model trained to perform radar target detection is proposed. Original loss functions that are not commonly used in deep learning-based model training for radar signal processing are introduced and their contributions to improvements on detection probability and false alarm control are highlighted.

The detection performances of the studied networks are determined on a singular target detection over exoclutter background scenario, which is the environment on which the performance gains are supposed to be minimal compared to the classical CFAR methods. It is shown that the neural detectors outperform the CFAR baseline in this setting. The next step, detection performance evaluation over more complex environments (presence of ground clutter, interferences, multiple targets, real data) will be the object of future research. Improvements of using neural detectors over CFAR detectors are expected to be much greater, as it has been noticed through early experiments.

References

[1] Louis L. Scharf and Cédric Demeure, *Statistical signal processing: detection, estimation, and time series analysis*, Addison-Wesley series in electrical and computer engineering. Digital signal processing. Addison-Wesley Pub. Co, Reading, Mass, 1991.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger, Eds. 2012, vol. 25, Curran Associates, Inc.

[3] Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," 2014.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep Residual Learning for Image Recognition," 2015.

[5] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully Convolutional Networks for Semantic Segmentation," 2014.

[6] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," 2015.

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, "Mask R-CNN," 2017.

[8] P.P. Gandhi and S.A. Kassam, "Analysis of CFAR processors in nonhomogeneous background," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 24, no. 4, pp. 427–445, July 1988.

[9] Chao Wang, Jian Wang, and Xudong Zhang, "Automatic radar waveform recognition based on time-frequency analysis and convolutional neural network," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, Mar. 2017, pp. 2437–2441, IEEE.

[10] Bharat Sehgal, Hanumant Singh Shekhawat, and Sumit Kumar Jana, "Automatic Target Recognition Using Recurrent Neural Networks," in *2019 International Conference on Range Technology (ICORT)*, Balasore, India, Feb. 2019, pp. 1–5, IEEE.

[11] Daniel Brodeski, Igal Bilik, and Raja Giryes, "Deep Radar Detector," in *2019 IEEE Radar Conference (RadarConf)*, Boston, MA, USA, Apr. 2019, pp. 1–6, IEEE.

[12] Li Wang, Jun Tang, and Qingmin Liao, "A Study on Radar Target Detection Based on Deep Neural Networks," *IEEE Sensors Letters*, vol. 3, no. 3, pp. 1–4, Mar. 2019.

[13] Faruk Yavuz, "Radar Target Detection with CNN," in *2021 29th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 1581–1585.

[14] Chenxing Wang, Jiangmin Tian, Jiuwen Cao, and Xiaohong Wang, "Deep Learning-Based UAV Detection in Pulse-Doppler Radar," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–12, 2022.

[15] Meiyan Pan, Jianjun Chen, Shengli Wang, and Ziwei Dong, "A Novel Approach for Marine Small Target Detection Based on Deep Learning," in *2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP)*, Wuxi, China, July 2019, pp. 395–399, IEEE.

[16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," 2015.

[17] Tzvi Diskin, Uri Okun, and Ami Wiesel, "Learning to Detect with Constant False Alarm Rate," 2022.

[18] Chia-Hung Lin, Yu-Chien Lin, Yue Bai, Wei-Ho Chung, Ta-Sung Lee, and Heikki Huttunen, "DL-CFAR: A Novel CFAR Target Detection Method Based on Deep Learning," in *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, 2019, pp. 1–6.

[19] Zachary Baird, Michael K. McDonald, Sreeraman Rajan, and Simon Lee, "A Neyman-Pearson Criterion-Based Neural Network Detector for Maritime Radar," in *2021 IEEE 24th International Conference on Information Fusion (FUSION)*, Sun City, South Africa, Nov. 2021, pp. 1–8, IEEE.

[20] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi, "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation," 2016.

[21] Seyed Sadegh Mohseni Salehi, Deniz Erdogmus, and Ali Gholipour, "Tversky loss function for image segmentation using 3D fully convolutional deep networks," 2017.

[22] Roberto Franceschi and Dmytro Rachkov, "Deep learning-based radar detector for complex automotive scenarios," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022, pp. 303–308.