2023 IEEE 10th International Conference on Data Science and Advanced Analytics (DSAA) | 979-8-3503-4503-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/DSAA60987.2023.10302539

A Continual Learning System with Self Domain Shift Adaptation for Fake News Detection

Sebastián Basterrech
Department of Applied Mathematics and Computer Science
Technical University of Denmark
Kongens Lyngby, Denmark
sebbas@dtu.dk
ORCID: 0000-0002-9172-0155

Andrzej Kasprzak
Department of Systems and Computer Networks
Wroclaw University of Science and Technology
Wroclaw, Poland
andrzej.kasprzak@pwr.edu.pl
ORCID: 0000-0003-3672-5006

Jan Platoš
Faculty of Electrical Engineering and Computer Science
VŠB-Technical University of Ostrava
Ostrava, Czech Republic
jan.platos@vsb.cz
ORCID: 0000-0002-8481-0136

Michał Woźniak
Department of Systems and Computer Networks
Wroclaw University of Science and Technology
Wroclaw, Poland
michal.wozniak@pwr.edu.pl
ORCID: 0000-0003-0146-4205

***Abstract*—Detecting fake news is currently one of the critical challenges facing modern societies. The problem is particularly relevant, as disinformation is readily used for political warfare but can also cause significant harm to the health of citizens, such as by promoting false data on the harmfulness of selected therapies. One way to combat disinformation is to treat fake news detection as a machine learning task. This paper presents such an approach, which additionally addresses an important problem related to the non-stationarity characteristics of the fake news. We elaborated a stream data with the simulation of domain shift based on two popular benchmark datasets dedicated to the fake news classification problem (Kaggle Fake News and Constraint@AAAI2021–COVID19 Fake News Detection). The proposed learning system works in a Continual Learning (CL) framework and integrates a self domain shift adaptation in a machine learning scheme. The method was built following state-of-the-art techniques, that includes Word2Vec as a feature extractor and the LSTM model as a classifier. The performance of the approach has been evaluated over the generated data stream. The convenience of our approach is showed in the results, where the accuracy gain with respect to a CL approach without domain adaptation is observed to be significant.**

***Index Terms*—Continual Learning, Domain Shift, Fake News, Disinformation, Word2Vec**

I. Introduction

Disinformation is one of the significant problems of democratic societies. It is an essential tool of political struggle that prevents citizens' objective perception of important events. Consequently, disinformation tools, such as disseminating so-called "fake news", can distort the results of elections, interfere with a fair assessment of the political situation, or negatively affect the reputation of public officials. This type of action could be observed in the last two presidential elections in Brazil and US, throughout the time of the COVID-19 pandemic, and has been widely used during the aggression of Russian Federation against Ukraine[1]. Therefore, developing tools for effective automatic fake news detection is a desirable direction for research using methods based on machine learning.

Unfortunately, most works consider the problem of detecting fake news as a stationary problem, not considering that those spreading disinformation know that automatic methods are used to detect it. Hence, the nature of fake news changes over time, and simultaneously, its subject changes due to emerging new topics. For instance, fake news related to the adverse effects of mRNA vaccines became extremely popular during the COVID-19 pandemic. As the pandemic expires, the global discussion about adverse effects of some vaccines becomes a marginal topic, giving way in popularity to the invasion of Russian Federation to Ukraine[2]. Thus, as can be seen, we are dealing with the problem of detecting fake news, in the course of which the model of the classification task is constantly evolving. Hence, it is necessary to propose models that can automatically adapt detectors to the changing nature of fake news. The answer to these needs is Continual Learning (CL). Such techniques should consider the possibility of changes in the data distribution, so-called concept drift. A particular case of distribution change is the domain adaptation problem, commonly referred to in the context of transfer learning techniques, where a model learns from data from a new distribution. We investigate the performance of a learning

This work was supported by the CEUS-UNISONO programme, which has received funding from the National Science Centre, Poland under grant agreement No. 2020/02/Y/ST6/00037, and the GACR-Czech Science Foundation project No. 21-33574K "Lifelong Machine Learning on Data Streams".

[1]United Nations General Assembly, the 11th emergency special session, 2022: https://digitallibrary.un.org/record/3966630?ln=en

[2]United Nations Meetings Coverage and Press Releases, March 2, 2022: https://press.un.org/en/2022/ga12407.doc.htm

system with domain adaptation for tackling the problem of fake news detection in non-stationary environments, which is a more realistic scenario for treating fake news detection. The most important contributions of this work are:

(i) Generation of data stream based on available public datasets that simulates the distribution changes in the context of fake news. We investigate the domain characteristics of the used datasets, as well as distribution changes in the stream, using projection pursuit visualizations.
(ii) Design of an efficient CL pipeline with self-domain shift adaptation. The proposed architecture has only three global parameters. Hence, we also include a fine-sensitive analysis of the parameter setting.
(iii) Exhaustive experimental evaluation of the proposed approach, and a comparative analysis with a baseline approach (the same system without integrating the domain shift detection).

This article is structured as follows. Section II presents a brief overview of the disinformation problem in the digital era. Section III describes the CL paradigm. Section IV introduces our proposed method. The following section provides the experimental setup and a discussion of our results. Finally, we summarize the learned concepts, our contributions, and discuss new directions for future works.

II. Fake news and misleading information in the social media era

Fake news has become one of the hottest topics of public discussion. Although it has gained prominence recently, it is not a new phenomenon. According to various experts [1], its origins date back to ancient times [2]. Disinformation was and still is an essential element of information war. However, it has gained importance in the era of easy access to often unverified content available through social media, which is the most popular information channel for reaching a large community. Unfortunately, deliberate disinformation is increasingly affecting the lives of societies, from using it for political warfare to spreading fake news about phenomena like the COVID-19 pandemic, or to using it to gain unwarranted market position by denigrating competitors or influencing consumer behavior. It is defining what fake news is a significant challenge. There are many definitions [3], including those published by a number of government agencies or international organizations. The European Union indicates that

> "Disinformation is false or misleading content that is spread with an intention to deceive or secure economic or political gain, and which may cause public harm. Misinformation is false or misleading content shared without harmful intent though the effects can be still harmful"[3].

There have been developed several methods to combat fake news [2], [4]. Some projects use the knowledge of human expertise to fight disinformation. Such an approach is particularly evident in the case of the fight against Russian disinformation, the precursors of which were the elf movement from Latvia, and now similar initiatives are emerging in other countries (the Czech Republic, Poland), where, due to Russian aggression against Ukraine, the activity of so-called trolls has increased. It is worth mentioning that such networks operate on a volunteer basis, and the goal is to identify sources spreading disinformation. However, the success of such initiatives strongly depends on legal regulations, such as allowing the possibility of periodic blocking of sites spreading disinformation. Another approach is using specialized websites so-called fact-checkers[4]. Such sites allow, for example, the presentation of multimedia content metadata so that anyone can verify its originality. Unfortunately, these solutions are limited in scope and do not allow for the evaluation of all emerging news. Therefore, more and more hope is placed on solutions based on machine learning techniques. Designing this type of system requires, foremost, gathering a reliably labeled learning set. This, in turn, requires an understanding of what fake news is, and it is crucial to distinguish an intentional hoax (actual fake news) from irony or satire that is close to it but completely different from its author's intention. It should also be pointed out that often the recognition of whether a given text is fake news may very much depend on an annotator's worldview, hence the process of preparing the data should be done with great care, to mitigate the bias produced by the subjective markup of the annotators. For instance, including a multilateral evaluation using different annotation schemes [5]. Although there have been successful collections of annotated fake news datasets in many areas, it should be taken into account that those who develop methods for spreading disinformation are aware of the characteristics of the detection techniques. It is said that good fake news is a little different from real information and conveys the content we expect. So, we need to be aware that the nature of information diffusion, which is a vehicle for disinformation, can evolve, so it is necessary to propose techniques capable of adapting fake news detection models to their changing characteristics.

III. The Continual Learning Paradigm

When considering the CL problem, we may divide problems into learning a single task or a group of tasks. During a single task learning, it is essential to indicate whether the considered task model is stationary. In this case, we only need to ensure that our algorithm can learn in incremental mode. Suppose we allow for changes in the model's parameters during operation. In that case, i.e., we are learning from non-stationary data streams, then we should use methods that can react to these changes, also known as concept drift [6]. In some cases, the problem could be also considered as a domain adaptation phenomenon.

When considering the process of learning a new predictive model, we often face the problem that a decision model may

[3]European Commission, Tackling online disinformation: https://digital-strategy.ec.europa.eu/en/policies/online-disinformation

[4]see, e.g., FactCheck.org in the US, or Demagog.org.pl in Poland

change while operating. Moreover, we cannot afford to collect data for a long time but must process it online. Furthermore, we often want to work with models that already have some minimum required quality, but we want to improve them during their operation. What is very important is that we have to take into account that when building such systems, we usually have limited resources, both computational and memory, which means that, on the one hand, we cannot store too large portions of data. On the other hand, we cannot afford to process the same object repeatedly. Finally, we may also face a significant distribution shift (concept drift) problem, which requires rebuilding and adapting the model to the new data distribution. It is worth noting that if we are dealing with a distribution shift and building a single-task learning model, as we have in this study, we need to provide mechanisms for adapting the model to the new distribution. Some models have built-in ability to adjust to the new parameters of the task. In contrast, others require additional mechanisms for forgetting data not from the new distribution so that these data do not negatively affect the newly built model.

Lifelong machine learning (LLML) proposes solutions that do not have the limitations typical of statistical learning algorithms, which require many learning examples and are tailored to learning single isolated tasks. A key mechanism in this approach is transfer learning, which allows using previously acquired knowledge. It is worth mentioning that LLML is inspired by human learning, which never learns in isolation or from scratch. Humans always retain knowledge acquired in the past and use it to learn new tasks. One of the main challenges facing LLML algorithms is the stability and plasticity dilemma. A model that is too stable cannot use new information from future training data. In contrast, a model with high plasticity allows significant parameter changes during learning, resulting in forgetting the previously learned representation (catastrophic forgetting). There are several taxonomies of continuous learning scenarios. Likely, the most popular one distinguished among [7]:

- **Task-incremental learning** assumes the model has access to task labels.
- **Domain-incremental learning** considers that the model does not have access to task labels, but all tasks involve the same classes. Models only need to be able to solve the task, but do not need to indicate which task it is.
- **Class-incremental learning** assumes, as in the previous scenarios, no access to task labels and that the classes are different in each task. So, models must be able to solve each task seen so far and infer which task they are presented with.

The algorithms developed by the community so far can also be divided into the following groups:

- **Regularization-based methods** modify the learning process to reduce forgetting. They assess how important the model parameters are for previously learned tasks and penalize future changes accordingly. Such algorithms preferentially train a different part of the network for each task, but always use the entire network for prediction. Elastic Weight Consolidation [8] introduces an additional regularization component that prevents learning for important parameters. In turn, Learning without Forgetting [9] uses pseudolabels derived from the classification of previous tasks to improve knowledge retention
- **Replay methods** are an alternative strategy for combating catastrophic forgetting. They are to complement the training data for each new task to be learned with "pseudo-data" representative of the previous tasks [10]. Deep Generative Replay [11] generates input samples paired with "hard targets" provided by the main modeliCaRL (incremental Classifier and Representation Learning) [12] uses a model to extract features and classifies based on the rule of the closest average class in that feature space, with class averages from memorized data. Gradient Episodic Memory (GEM) [13] uses memory samples to project a gradient that does not increase losses for previous tasks.
- **Methods based on the network architecture adaptation** propose extending the network structure to encompass changes in the data distribution. A typical representative of this approach is Progressive Neural Networks [14], which adds new backbone layers connected to previous layers to take advantage of knowledge learned from previous tasks.

The proposed approach in our study belongs to the domain-incremental learning. If we consider the second discussed taxonomic criterion, our approach belongs to the group that mitigates the forgetting [7].

IV. Methodology

Let us assume that data come in a sequence of ordered chunks $S_0, S_1, \ldots, S_k, \ldots$ over a discrete time indexed in the natural numbers. Each chunk is composed by a sequence of multidimensional labeled samples indexed over discrete time instants $t = 0, 1, \ldots, T$. In other words, each chunk S_i has input-output observations $\left(\mathbf{z}_i^{(1)}, \ldots, \mathbf{z}_i^{(K)}\right)$, with $\mathbf{z}_i^{(k)} = \left(\mathbf{u}_i^{(k)}, \mathbf{y}_i^{(k)}\right)$. The input-output pair is defined over the domain $\mathbb{U} \times \mathbb{Y}$, where $\mathbb{U}$ is a p-dimensional real space and $\mathbb{Y}$ is a set with the encoded classes. We are addressing the fake news detection problem, then in our experiments we only consider the situation $\mathbb{Y} = \{0, 1\}$. The goal in conventional learning is to train a parametric predictor by minimizing a loss function $\mathcal{L}(\cdot)$ using data from an arbitrary data window.

A. Domain shift detection problem

Domain shift, so-called distribution shift, refers to the change in the data distribution over time. We assume that the output space is fixed: we work in a binary classification context (fake or not fake); and the input distribution may constantly evolve. By the input distribution, we refer to the distribution of the representation of the digital news. As we mentioned earlier, news format presentation and news generation are constantly changing. There are frequent changes in the trending topics, many categories of news, some news are seasonal, and other topics are specific to single events.

Also, the news are correlated to external variables such as age groups, mobile applications, regions, and cultural aspects. Furthermore, mobile applications are also dynamic. Each mainstream media and mobile application share information using different information representations (different emoticons, number of words, pictures, and other attributes). Thus, when designing a robust fake news detection, it is necessary to consider the domain shift problem.

The domain shift may occur either by a change in the posterior distribution $\big(\Pr(\mathbf{y} \mid \mathbf{u})\big)$ or by a change in the prior distribution (distribution of $\mathbf{u}$) [15]. A reasonable example of a change in the posterior distribution is the situation that considers certain news labeled as fake news until a particular time step. After a breaking point, similar news related to this topic are not considered as fake news anymore. There are many situations like this in the history. An example, that even comes from the root of the modern sciences, it is the moment that Galileo Galilei announced his preference for the Copernican view "the earth orbits the sun instead of the idea that the earth is the center of the universe". This idea was taken as fictitious for a long time. Even some materials produced by Galileo Galilei were banned for almost two centuries after his dead in important European regions. There are also many examples on the opposite direction, i.e., situations where the news are considered real until a certain moment where a breaking point occurs, and additional evidence shows that the news were intentionally fake creations. For instance, examples of governmental institutions that used false claims for mass manipulation, or for attacking their political opponents.

Let us assume that there exists a Markov kernel process from $\mathcal{T}$ to $\mathbb{U}\times\mathbb{Y}$ that associates a distribution p_t on $\mathbb{U}\times\mathbb{Y}$ with every time point $t \in \mathcal{T}$ such that $t \mapsto p_t(A)$ is a measurable map for all measurable set $A \subset \mathbb{U}\times\mathbb{Y}$ [16]. The domain shift problem consists in identifying time points t such that the data distribution until t ($\{\ldots, \mathbf{z}_{t-2}, \mathbf{z}_{t-1}, \mathbf{z}_t\}$) is different from a future data distribution ($\{\mathbf{z}_{t+\Delta}, \mathbf{z}_{t+\Delta+1}, \ldots\}$, for $\Delta > 0$) [17]. It may also be expressed as follows, the shift detection problem consists of identifying all-time points t such that p_t and $p_{t+\Delta}$ differ for some small interval $\Delta > 0$. The magnitude of such difference can be estimated using a dissimilarity metric between the underlying distributions $dist(p_t, p_{t+\Delta})$, where $dist(\cdot)$ is an arbitrary-selected dissimilarity metric [16]. We apply a peak over threshold criterion for deciding when the dissimilitude is significant. Let's denote the threshold for controlling distribution shifts as θ_1.

B. Classifier selection in the context of domain shift adaptation

Let us describe a decision-making procedure when a domain shift is detected. Once a data distribution shift occurs, it is necessary to update the predictor, or even in some applications, it is required to reset the old one and create a new predictor. We designed a procedure with a competitive learning phase inspired from Kohonen's map [18] and scale invariant maps [19]. In the competitive learning step, the most suitable single predictor is selected. We call this predictor as: *Best Matching Model (BMM)*. Let's assume that, at the time t we have already learned M classifiers $\phi_1(\cdot), \phi_2(\cdot), \ldots, \phi_M(\cdot)$. We assign for each classifier $\phi_i(\cdot)$ a mass center $\mathbf{c}_i$, $\mathbf{c}_i \in \mathbb{U}$. The mass center can be interpreted as a representative data point or a cluster center. Let $\mathbf{c}_i$ be the mass center that characterizes the training data used for tuning the parameters of $\phi_i(\cdot)$

$$\mathbf{c}_i = \frac{1}{K}\sum_{k=0}^{K} \mathbf{u}_i^{(k)}, \tag{1}$$

where K denotes the number of samples. The BMM is defined as the model with an associated mass center closest to the presented input data. Given the current chunk S_i with the input data $\mathbf{u}_i = (\mathbf{u}_i^{(1)}, \ldots, \mathbf{u}_i^{(K)})$, then we define the BMM $\phi_{\mathrm{BMM}}(\cdot)$ as the model with the mass center that minimizes

$$\min_m \sum_{k=0}^{K} dist\big(\mathbf{c}_m, \mathbf{u}_i^{(k)}\big), \tag{2}$$

with $m \in [1, M]$. As we mentioned above, the function *dist*$(\cdot,\cdot)$ is a selected measure of dissimilitude between two multidimensional points. Note that the previous definition requires that at least exists one learned model. Therefore, at the initial step of the proposed approach, we define $\mathbf{c}_0$ as the mass center for the first chunk, and $\phi_{\mathrm{BMM}}(\cdot) = \phi_0(\cdot)$ as the model that is learned using the data in the chunk S_0.

Furthermore, we assume the presence of a non-stationary environment. Therefore, the distance between the mass centers and the new data may be too large, then a new model needs to be learned. However, the capacity for storing new classifiers is finite. As a consequence, we define a decision rule with a threshold parameter θ_2, which specifies if either is convenient to learn a new model or not. Given the chunk S_i, if the expression $\sum_{k=0}^{K} dist\big(\mathbf{c}_m, \mathbf{u}_i^{(k)}\big) > \theta_2$ is satisfied, then a new model $\phi_j(\cdot)$ is learned with the current input data. In addition, a new mass center $\mathbf{c}_j$ is computed.

C. Workflow of the proposed approach

Figure 1 introduces the CL pipeline of the proposed system. The rectangular nodes in the diagram represent the objects: data input, a set of learning models, and the model prediction data. There are also ellipsoid nodes, which depict the following two decision processes and one standard classification algorithm:

(i) Domain shift detection: it is implemented according to the description presented in section IV-A. In case a shift is decided, the model selection is performed. Otherwise, the input data is used for online training and producing the model predictions.
(ii) Model selection: the selection of the classifier is made according to the description presented in section IV-B. In the diagram, the model selection node also includes generating new models.
(iii) Learning process: it refers to a CL training procedure.

The global approach presented in Figure 1 has three hyperparameters: the capacity for storing the models, that we refer

by *model buffer size*, a *dissimilarity threshold* for detecting domain shifts, and a *model accuracy threshold* for deciding when is a correct moment to switch models (or to create a new one). The model buffer, that we denote from now as M, it is the size of the collection where are stored the learned models. Due to memory restrictions, we assume a specific limited buffer size. We compute the dissimilarity between vectors for detecting domain shifts. Then, we decided to apply the strategy of peak over the threshold, i.e., in the case of computing a dissimilarity larger than the threshold θ_1. We recognize the occurrence of a distribution shift in the input domain. Furthermore, we use a threshold for deciding when to learn a new model or switch to another one in the buffer. We learn a new model when the obtained accuracy exceeds a tolerance threshold θ_2.

For a better readability, we split the proposed method into two pseudocodes. Algorithm 2 introduces the pseudocode of the global system. In line 11, Algorithm 2 invokes Algorithm 1. This second pseudocode (Algorithm 1) describes the procedure for selecting an appropriate predictor, in order of processing the current input chunk. The sentences between line 19 and line 22 describe the actions to be performed when the model buffer is full. In the empirical experiments, we treat the buffer as a FIFO queue. When the buffer is full, we replace the oldest model for a new model. However, there are other possible strategies, e.g., to define a priority queue where the priority weights are related to the frequent usage of the model. A model that is more often used will have associated a higher priority than a model that is used with a low frequency. Also, it is possible to make a relationship between the accuracy and the priority of being selected. In the experimental section, we evaluated only the simplest situation where the queue is FIFO, then when the buffer is full the oldest model in the queue leaves space in the buffer for a new learned model.

D. Metric selection

A crucial step of the proposed approach is selecting the model to be used once a distribution shift is detected. This step is performed by computing the mass centers with the input data, i.e. a dissimilarity score among high dimensional points is computed. Euclidean distance is sensitive to noise, suffers when the data is sparse, and has limitations in high dimensional spaces [20]. Similar disadvantages are presented in the Manhattan distance. Therefore, in the experimental results, we decided to use the Hellinger distance that is more suitable for working with high dimensional data [21]. In addition, the functional value is bounded in $[0, \sqrt{2}]$. Hellinger distance is defined for quantifying probability distributions. Therefore, before computing the distance, we normalize the data in $[0, 1]$. Given two p-vectors $\mathbf{c}$ and $\mathbf{u}$, the selected distance has the form [22]:

$$\mathrm{H}(\mathbf{c}, \mathbf{u}) = \left(\sum_{i=0}^{p} \left(\sqrt{c(i)} - \sqrt{u(i)} \right)^2 \right)^{1/2}, \qquad (3)$$

where $c(i)$ and $u(i)$ denote the coordinate i of the vector $\mathbf{c}$ and $\mathbf{u}$.

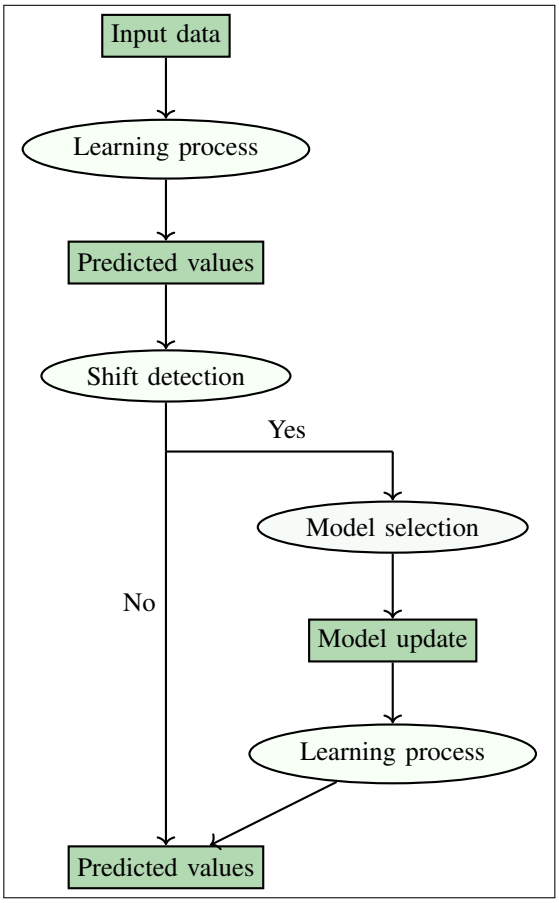

Fig. 1: Top-down workflow of the proposed approach.

Let's notice that another distance function is used to decide whether a distribution shift has occurred or not. In this case, we apply the difference between two scalars. In the experimental section, it is used the Euclidean distance. Furthermore, Kullback–Leibler divergence is used for training the classifier, which is commonly employed at the moment of learning binary predictors [23].

V. Experimental evaluation

The experimental research aims to demonstrate the utility and relevance of applying domain shift adaptation in a CL framework for detecting fake news. The experiments intend to answer the following research questions:

- RQ1: Does implementing a domain shift detector positively affect the quality of a fake news classifier?
- RQ2: How does the proposed detector parameters setting impact its quality?

During the experiments, we examine the characteristics of the input domain using visualization tools (non-linear projections into the 3D space). We also present the accuracy obtained in offline contexts without artificially injected domain shifts. The reached accuracy in offline contexts can be seen as an upper bound (a reference score) of the accuracy reached in a non-stationary environment. Besides, we used different

Algorithm 1: High level description of the module selection procedure.

Input: Set of mass centers ($\mathcal{C}$), collection of models (Φ), chunk data (S_i), threshold used for model selection (θ_2)
Result: Selected model $\left(\phi_{BBM}(\cdot)\right)$, set of mass centers ($\mathcal{C}$), collection of models (Φ)

// Select BMM
$distance = zeros(array, M)$;
// For each mass center $\mathbf{c}_m$, $\mathbf{c}_m \in \mathcal{C}$
for $m = 1$ *to* M **do**
  $cumDist = 0$;
  // For each sample $\mathbf{u}_i^{(k)}$, $\mathbf{u}_i^{(k)} \in S_i$
  for $k = 1$ *to* K **do**
    $cumDist = cumDist + dist(\mathbf{u}_i^{(k)}, \mathbf{c}_i)$;
  end for
  $distance[m] = cumDist$;
end for
// minimum value between mass centers and the input data
$distMin = min(distance)$;
// index of the minimum value
$BMM = argmin(distance)$;
if *(*$distMin > \theta_2$*)* **then**
  // Train a new model with data S_i
  $\phi_{new}(\cdot) = createNewModel(\cdot)$;
  if $|\Phi| < M$ **then**
    Add $\phi_{new}(\cdot)$ into collection Φ;
    $\mathbf{c}_{new} = computeMassCenter(S_i)$;
    Add $\mathbf{c}_{new}$ into set $\mathcal{C}$;
  end if
  else
    $\phi_{old}(\cdot) = selectOldClassifier(\Phi)$;
    // Update collection
    Replace $\phi_{old}(\cdot)$ for $\phi_{new}(\cdot)$ in collection Φ;
    $\mathbf{c}_{new} = computeMassCenter(S_i)$;
    Add $\mathbf{c}_{new}$ in $\mathcal{C}$;
  end if
  // Last computed model $\phi_{new}(\cdot)$ is assigned to be the BMM
  Update BMM;
end if
return $\phi_{BBM}(\cdot)$, Φ, $\mathcal{C}$;

Algorithm 2: High level description of the proposed machine learning pipeline.

Input: Maximum number of saved models (M), threshold used for shift detection (θ_1), threshold used for model selection (θ_2), parameters for model creation, streaming data
Result: Collection of models (Φ), predicted values ($\hat{\mathbf{y}}$)

Obtain chunk S_0;
$\phi_0(\cdot) = createNewModel()$;
Compute predicted values $\mathbf{y}_0^{(k)}$, $k \in [1, K]$;
Add $\phi_0(\cdot)$ into Φ;
$\mathbf{c}_0 = computeMassCenter(S_0)$;
Add $\mathbf{c}_{new}$ into $\mathcal{C}$;
while *(it arrives* S_i*)* **do**
  $\phi_i(\cdot) = createNewModel(\cdot)$;
  Compute $\hat{\mathbf{y}}_i^{(k)}$, $k \in [1, K]$;
  if *(*$shiftDetection(S_i, \mathcal{C}, \theta_1) == YES$*)* **then**
    // Apply Algorithm (1)
    $\phi_{new} = selectModel(\mathcal{C}, \Phi, S_i, \theta_2)$;
    Compute $\hat{\mathbf{y}}_i^{(k)}$, $k \in [1, K]$;
  end if
end while
return Φ, predicted values $\hat{\mathbf{y}}$;

hyperparameters in our machine learning pipeline to analyze the sensitivity of the model's primary hyperparameters.

A. Setup

Datasets. During the experiments, we used the following well-known datasets:

1) *Kaggle Fake News dataset* [24]. It is a balanced binary dataset composed of labeled news as real and fake news (a total of 26000 samples). The real news was selected from many popular United States mainstream news websites. The fake news were automatically created based on real news using Mechanical Turk [25], [26].
2) *Constraint@AAAI2021 - COVID19 Fake News Detection in English* [27]. This benchmark problem was introduced in a Workshop in AAAI, 2021. The data has 5600 real news and 5100 fake news. The 10700 posts were manually annotated from social media posts and articles with news related to COVID-19.

In Table I, we present five randomly taken samples from the two used datasets to illustrate the similarity between both sources of information.

Streaming data processing. The data stream for the experiments was created as follows. We split the Kaggle Fake News into four subsets (with randomly selected samples) and the Covid19 Fake News into three subsets. Then, we create the stream interchanging the subsets from one set and another set. Figure 2 depicts how the data stream was generated. Therefore, the construction of the data stream was done in a way for simulating sudden recurring concept drifts. The changes between data chunks from one dataset and chunks from another dataset simulate a domain change in the fake news characteristics.

Data preprocessing. The stemming process consisted of changing to lowercase, and we removed morphological affixes from words using the *nltk.stem.porter* library. We computed the embedding vectors following the solution studied [28]. The authors used Word2Vec and LSTM to obtain a high rank in the competition among the offline classification solutions. We fixed the embedding dimension to 100. We combined both

TABLE I: Examples of fake news, in order of showing the similarity between samples of the used datasets.

Covid19 dataset	Real	*Coronavirusupd india fights corona more than lakh test done for rd success day cumul test as on date ha reach covid test per million tpm cross.*
Covid19 dataset	Fake	*Scientist at astrazeneca complain their work on a coronaviru vaccin keep be delay by noddi holder ring up to ask if it will be readi by christma http t co lmyztnapx.*
Covid19 dataset	Fake	*cdc recommend mother stop breastfeed to boost vaccin efficaci*
Kaggle dataset	Real	*new guidelines from the National Institute of Allergy and Infectious Diseases that advise giving peanuts to children early and often as a strategy to forestall an allergy.*
Kaggle dataset	Fake	*The fatalities have seemingly occurred after a breakthrough cancer treatment was just announced.*

collections of news (from Kaggle and Constraint@AAAI). Afterward, we applied the mapping from words to vectors using the Word2Vec technique [29]. The text tokenization was done using the TensorFlow Keras library with a maximum length of padding sequences of 700.

Learning procedure. Each data chunk consisted of 500 samples. The selected classifier was the LSTM model [30], due to the achieved performance in [28]. The LSTM architecture was implemented in Keras with the following characteristics: sigmoid and relu activations, Adam optimization algorithm, dropout generalization (parameter equal to 0.1), and cross-entropy as a loss function. A first embedding layer, two hidden layers (with 128 neurons), a dropout layer, and a dense readout layer.

Implementation details. The proposed method was implemented in Python programming language *Python 3.8.16* with import popular libraries for Data Science operation as: *Numpy*, *Pandas*, *Tensor Flow*, *Sklearn*. In addition, we used *Skmultiflow* library for creating and manipulating data streams. We used *Nltk* library to perform the steaming process and obtain the corpus for the embedding vector computation.

B. Analysis of the input space

In the following, we investigate the characteristics of the input space for the created data stream. We investigate the learning model's input space to check if there are domain shifts in the data stream. Projection pursuit is a family of techniques that share the basic idea of analyzing the data in a lower dimensionality space instead of making the data analysis directly in the original space. The most popular technique of this type is probably PCA [19]. Here, we use another popular technique named t-distributed Stochastic Neighbor Embedding (t-SNE), a widely used non-linear projection for visualizing multidimensional data in a three-dimensional space [31]. Figure 3 depicts the t-SNE projection of the created data stream. The visualization suggests that a domain shift occurs in the data stream, which is relevant for our experiments. Note that, the projection depicts intrinsic geometric characteristics among the points, therefore we explicitly present the figure coordinate-free. The relationship among the projected points of both dataset doesn't depend on any arbitrary chosen coordinates. The cloud of points has plots of four types of dots: blue and red dots belong to Kaggle Fake News, and green and orange belong to the Covid19 dataset. Blue and green represent real news, and red and orange represent fake news. Note that the projection of the Covid dataset seems to belong to a different domain than the projection of the Fake News dataset. For this example, the effect of domain shift can be mitigated by rescaling the input data [32]. However, we do not transform both datasets (Kaggle Fake News and Covid-19) to the same domain to ensure that the domain shift occurs.

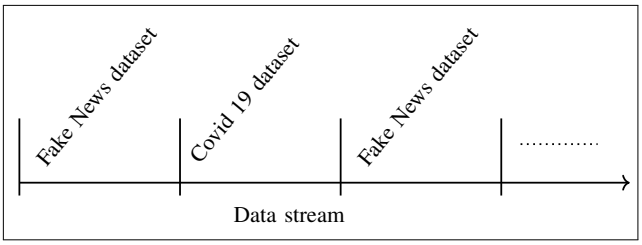

Fig. 2: Visualization of the created stream data, we interexchanged samples from both datasets.

C. Result analysis

Firstly, we compute the accuracy of the learning model in an offline context. Figure 4 shows the accuracy value for different epochs. The approach is fully offline, then the batch size is equal to the size of the training dataset. The figure also presents a 95% t-student confidence interval of 10 experiments with different random initialization of the learning tool. We use that reached accuracy as a reference assessment. Figure 5 shows the impact of the proposed self-adaptation. The blue curve presents the results over the generated dataset (a combination of the two benchmark datasets). A contour around the curve shows the 95% confidence interval of the accuracy. The blue curve presents an example of results for the proposed approach with thresholds $\theta_1 = 0.08$ and $\theta_2 = 11$. It is visible how the proposed approach can mitigate the degradation provoked by the domain shifts. Once the domain shift is detected, the method can switch to a more suitable classifier. We analyzed the effect of the two hyperparameters. Two plots show how impacts θ_1 with a fixed θ_2 value according to the chunk arrivals. The threshold used for drift was evaluated for the values of $\{0.04, 0.05, 0.06, 0.07, 0.08, 0.09\}$. The threshold used for the model selection was evaluated for the values $\{7, 8, 9, 10, 10.5, 11, 12, 12.5\}$. Figure 6 presents the results computed with $\theta_2 = 11$ (tolerance threshold for changing the model), and Figure 7 presents the results computed with

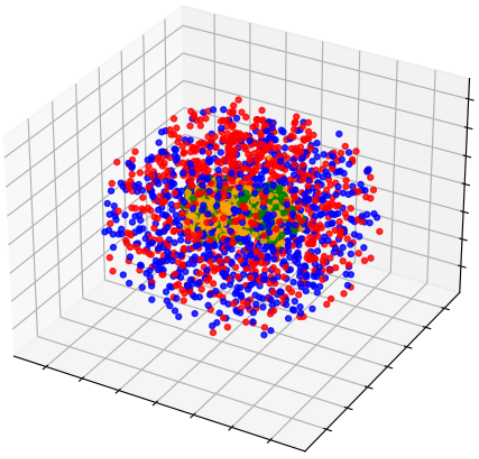

Fig. 3: A t-SNE visualization for showing similitude between both datasets (testing samples). Both datasets are mixed, red dots represent the fake news of Kaggle dataset and orange dots represent the Covid fake news. Blue dots and green dots represent the real news on Kaggle dataset and Covid dataset, respectively. For the Kaggle dataset is much larger, therefore we randomly selected points in order of having the 50% of dots from Kaggle and 50% from Covid dataset.

a threshold $\theta_2 = 12$. For both figures, the horizontal axis shows the time (chunk arrivals), and the vertical axis has the threshold θ_1 values. Furthermore, Figure 8 shows the impact of the parameters $\theta_2 = 11$ and $\theta_2 = 12$. Figure 8 presents results aggregating the accuracy and variance for the whole stream. There are two charts that present the averaged accuracy according to four different values of θ_1 and θ_2 equal to 11 and 12, and two charts that shows the accuracy dispersion (standard deviation), also for different values of θ_1 and θ_2 equal to 11 and 12.

D. Discussion

The essence of the introduced method is based on two decisions: (i) how to detect the domain shift?, and (ii) what to do once the shift is detected? The parameter θ_1 affects the decision (i), and the other parameter θ_2 impacts the decision (ii). Note that a large threshold θ_1 will make the system unable to identify domain shifts, then the system will behave as a classic CL procedure without adaptation. On the other hand, a small value of θ_1 may lead the system to fall in misclassifying false positive drifts. Recognition of false positives has another negative consequence, recall that at each detected drift, the model switches the predictor to another model. Therefore, a false positive may provoke the system to decrease knowledge consolidation in the CL procedure. The parameter θ_2 also affects the capacity of the model to consolidate the learned knowledge. A lower threshold θ_2 implies that the system will start learning a new model every time a drift is detected. Thus, knowledge consolidation is negatively affected. Conversely, a larger value of θ_2 will ensure that the proposed method always selects among the models in the buffer. Then, even if the new data has a different distribution, an old model will be selected and used as an initial model for the training with the latest data. In the experimental section, the estimation of both threshold values has been done empirically. We used initial time windows and evaluated the threshold using a grid search.

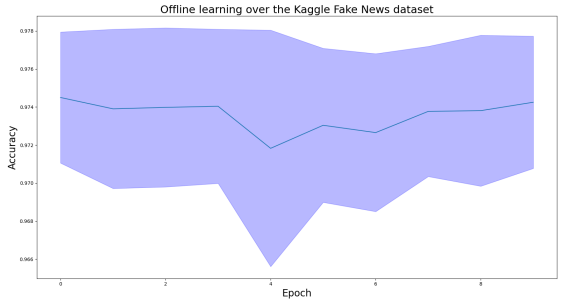

Fig. 4: Validation accuracy using Word2Vec + LSTM in the Fake news dataset. The training was done offline over the training dataset of the competition. The 95% confidence interval was made over the results of 10 different experimental trials of the LSTM model.

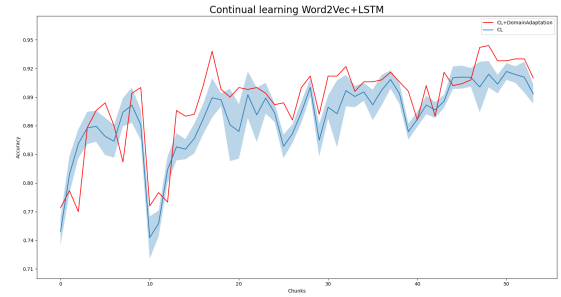

Fig. 5: Example of obtained accuracy with continual learning and domain adaptation over the simulated data stream. The red curve was obtained with a model with drift threshold value 0.08 and model threshold 11. The blue curve shows the results when continual learning without domain control is performed. A 95% confidence interval was made over the results of 20 different experimental trials of the Word2Vec+LSTM model (without domain shift control).

VI. Lessons Learned

Based on the obtained results of the experiments, we answer the initial research questions in the following. The experiments showed that detecting the distribution shift in a cognizant way improves the classification quality, which is particularly well

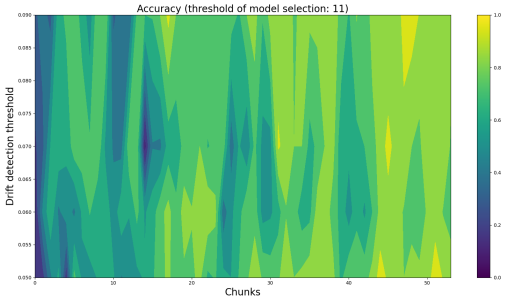

Fig. 6: Evolution of the model accuracy: impact of the drift detector parameter (θ_1) when the threshold $\theta_2 = 11$.

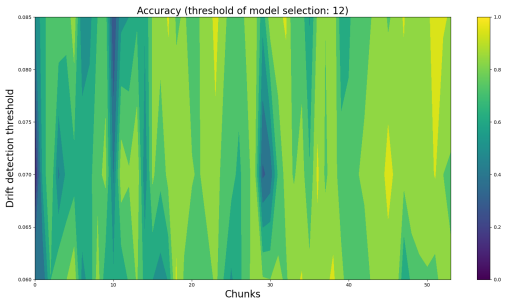

Fig. 7: Evolution of the model accuracy: impact of the drift detector parameter (θ_1) when the threshold $\theta_2 = 12$.

demonstrated in Figure 5. It is also worth noting that the approach ensures that the decrease in classification quality when a domain drift occurs is less severe than when we do not use an appropriate detection method. This is an important characteristic that is the subject of an analysis proposed by Shaker and Hüllermeier [33], where two performance metrics have been proposed: restoration time and maximum performance loss (RQ1 answered)).

Furthermore, the proposed method is parametrized, and the selection of appropriate parameter values, i.e., the buffer size and the two thresholds, significantly impact the prediction quality of the resulting model. Hellinger distance is bounded, therefore it has been helpful at the moment of selecting the parameter θ_2. In our experiments, we have settled the buffer size equal to 5. In future studies, we can use a subset of news and make clusters (using unsupervised techniques), then a more appropriate model size can be used as the number of clusters (RQ2 answered). Hence, when applying the proposed approach to other classification tasks, careful hyperparameter tuning should be made, using, for example, SMAC (Sequential Model Algorithm Configuration) [34] that employs *Bayesian Optimization* and aggressive racing mechanism to optimize parameters.

VII. Conclusion and Future Works

This work presented how to combat disinformation using machine learning methods. Additionally, we considered the

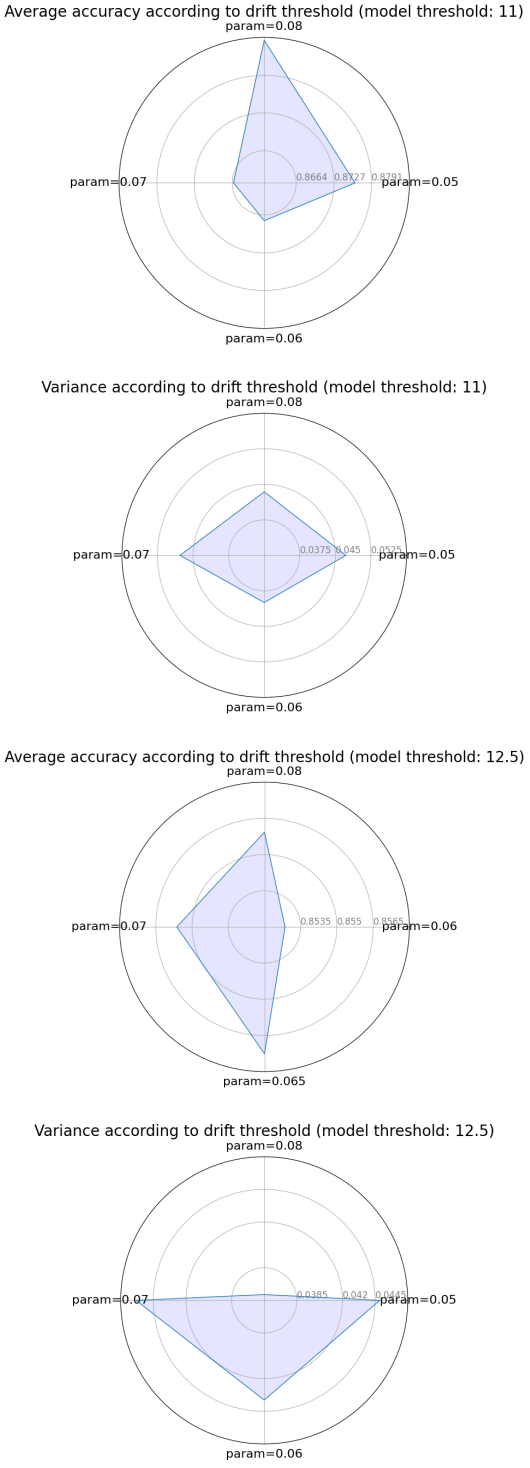

Fig. 8: Charts show the average and standard deviation of the reached accuracy for different values of the parameters θ_1 and θ_2.

significant problem related to the non-stationarity of the fake news classification model. Supported by statistical confidence intervals, we showed the usefulness of the proposed method, which was proven through a series of computer experiments using a generated data stream. The proposed methods employed a continual learning approach and integrated a self-domain adaptation in a learning pipeline. To evaluate this proposition, we conducted experiments on an artificially created data stream that simulates input domain changes. We showed that continual learning with self-domain adaptation obtained a significantly higher accuracy than a system without adaptation. We advanced state-of-the-art on self-adaptive domain shift for fake news detection task.

In the near future, we plan to address the following research challenges: (i) hyperparameter tuning: evolutionary algorithms or Bayesian optimization can be integrated for a better hyperparameter selection; (ii) text representation: we expect to explore the potential of transformers-based techniques instead of Word2vec; (iii) improving the classification task: we plan to incorporate different architectural layouts and classifier ensemble approaches.

REFERENCES

[1] J. Posetti and A. Matthews, "A short guide to the history of'fake news' and disinformation," *International Center for Journalists*, vol. 7, 2018.

[2] M. Choraś, K. Demestichas, A. Giełczyk, Álvaro Herrero, P. Ksieniewicz, K. Remoundou, D. Urda, and M. Woźniak, "Advanced machine learning techniques for fake news (online disinformation) detection: A systematic mapping study," *Applied Soft Computing*, vol. 101, p. 107050, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494620309881

[3] G. D. Domenico, J. Sit, A. Ishizaka, and D. Nunan, "Fake news, social media and marketing: A systematic review," *Journal of Business Research*, vol. 124, pp. 329–341, 2021.

[4] X. Zhang and A. A. Ghorbani, "An overview of online fake news: Characterization, detection, and discussion," *Information Processing & Management*, vol. 57, no. 2, p. 102025, 2020.

[5] R. Pandey, H. Purohit, C. Castillo, and V. L. Shalin, "Modeling and mitigating human annotation errors to design efficient stream processing systems with human-in-the-loop machine learning," *International Journal of Human-Computer Studies*, vol. 160, p. 102772, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1071581922000015

[6] B. Krawczyk, L. L. Minku, J. ao Gama, J. Stefanowski, and M. Woźniak, "Ensemble learning for data stream analysis: A survey," *Information Fusion*, vol. 37, pp. 132 – 156, 2017.

[7] G. van de Ven, T. Tuytelaars, and A. Tolias, "Three types of incremental learning," *Nature Machine Intelligence*, vol. 4, pp. 1–13, 12 2022.

[8] J. Kirkpatrick, R. Pascanu, N. C. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *CoRR*, vol. abs/1612.00796, 2016. [Online]. Available: http://arxiv.org/abs/1612.00796

[9] Z. Li and D. Hoiem, "Learning without forgetting," *CoRR*, vol. abs/1606.09282, 2016. [Online]. Available: http://arxiv.org/abs/1606.09282

[10] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. K. Dokania, P. H. S. Torr, and M. Ranzato, "Continual learning with tiny episodic memories," *CoRR*, vol. abs/1902.10486, 2019. [Online]. Available: http://arxiv.org/abs/1902.10486

[11] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," 2017.

[12] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[13] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continuum learning," *CoRR*, vol. abs/1706.08840, 2017. [Online]. Available: http://arxiv.org/abs/1706.08840

[14] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *CoRR*, vol. abs/1606.04671, 2016. [Online]. Available: http://arxiv.org/abs/1606.04671

[15] I. Goldenberg and G. I. Webb, "Survey of distance measures for quantifying concept drift and shift in numeric data," *Knowledge and Information Systems*, vol. 60, pp. 591–615, 2019.

[16] F. Hinder, V. Vaquet, and B. Hammer, "Suitability of Different Metric Choices for Concept Drift Detection," in *Advances in Intelligent Data Analysis XX*, T. Bouadi, E. Fromont, and E. Hüllermeier, Eds. Cham: Springer International Publishing, 2022, pp. 157–170.

[17] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys*, vol. 46, no. 4, pp. 44:1–44:37, 2014. [Online]. Available: http://doi.acm.org/10.1145/2523813

[18] T. Kohonen, *Self-Organizing Maps*, 3rd ed. Springer Series in Information Sciences, 2001, vol. 30.

[19] C. Fyfe, *Hebbian Learning and Negative Feedback Networks*, 1st ed., ser. Advanced Information and Knowledge Processing. Springer-Verlag London, 2005, vol. XVIII.

[20] A. Zimek, E. Schubert, and H. Kriegel, "A survey on unsupervised outlier detection in high-dimensional numerical data," *Statistical Analisys and Data Mining*, vol. 5, pp. 363–387, 2012.

[21] G. Ditzler and R. Polikar, "Hellinger distance based drift detection for nonstationary environments," in *IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE)*, 2011, pp. 41–48.

[22] A. L. Gibbs and F. E. Su, "On choosing and bounding probability metrics," *arXiv, available: https://arxiv.org/abs/math/0209021*, 2002. [Online]. Available: https://arxiv.org/abs/math/0209021

[23] S. Basterrech and M. Woźniak, "Tracking changes using Kullback-Leibler divergence for the continual learning," in *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2022, pp. 3279–3285.

[24] W. Lifferth, "Fake news," 2018. [Online]. Available: https://kaggle.com/competitions/fake-news

[25] S. S. Ahmed H, Traore I, "Detection of online fake news using n-gram analysis and machine learning techniques," in *Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments (ISDDC'2017)*. Lecture Notes in Computer Science, Springer, 2017, pp. 127–138.

[26] ——, "Detecting opinion spams and fake news using text classification," *Journal of Security and Privacy*, vol. 1, 2018.

[27] P. Patwa, M. Bhardwaj, V. Guptha, G. Kumari, S. Sharma, S. PYKL, A. Das, A. Ekbal, S. Akhtar, and T. Chakraborty, "Overview of CONSTRAINT 2021 Shared Tasks: Detecting English COVID-19 Fake News and Hindi Hostile Posts," in *Proceedings of the First Workshop on Combating Online Hostile Posts in Regional Languages during Emergency Situation (CONSTRAINT)*. Springer, 2021.

[28] Z. Samchuk, "Competation notebook kaggle fake news. solution proposed by user zhenddos entitled "word2vec+lstm"," url: https://www.kaggle.com/code/zhenddos/word2vec-lstm-96-accuracy/notebook.

[29] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," 2013, url: https://arxiv.org/abs/1301.3781.

[30] S. Hochreiter and J. Schmidhuber, "Long Short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[31] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008. [Online]. Available: http://www.jmlr.org/papers/v9/vandermaaten08a.html

[32] B. Li, A. Drozd, Y. Guo, T. Liu, S. Matsuoka, and X. Du, "Scaling Word2Vec on Big Corpus," *Data Science and Engineering*, vol. 4, pp. 157–175, 2019.

[33] A. Shaker and E. Hüllermeier, "Recovery analysis for adaptive learning from non-stationary data streams: Experimental design and case study," *Neurocomputing*, vol. 150, pp. 250–264, 2015.

[34] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Learning and Intelligent Optimization*, C. A. C. Coello, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 507–523.