Original article

Integrated Computer-Aided Engineering
2025, Vol. 0(0) 1–17

DOI: 10.1177/10692509241306580
journals.sagepub.com/home/ico

Identification and explanation of disinformation in wiki data streams

Francisco de Arriba-Pérez[1], Silvia García-Méndez[1], Fátima Leal[2], Benedita Malheiro[3,4] and Juan C Burguillo[1]

Abstract

Social media platforms, increasingly used as news sources for varied data analytics, have transformed how information is generated and disseminated. However, the unverified nature of this content raises concerns about trustworthiness and accuracy, potentially negatively impacting readers' critical judgment due to disinformation. This work aims to contribute to the automatic data quality validation field, addressing the rapid growth of online content on wiki pages. Our scalable solution includes stream-based data processing with feature engineering, feature analysis and selection, stream-based classification, and real-time explanation of prediction outcomes. The explainability dashboard is designed for the general public, who may need more specialized knowledge to interpret the model's prediction. Experimental results on two datasets attain approximately 90% values across all evaluation metrics, demonstrating robust and competitive performance compared to works in the literature. In summary, the system assists editors by reducing their effort and time in detecting disinformation.

Keywords

Disinformation, eXplainable artificial intelligence, machine learning, natural language processing, stream processing, wikis

Received: 9 October 2024; accepted: 24 November 2024

1 Introduction

Alternative news sources, such as popular social media platforms, have changed how information is generated and disseminated. These sources of unverified content constitute a social concern regarding media trustworthiness and data accuracy, negatively impacting reader critical judgment.[1] In this regard, recent advances in Natural Language Processing (NLP) like Large Language Models (LLMs) have made the generation of human-like web content more affordable, accessible, and faster.[2] Unfortunately, the latter phenomenon can contribute to disinformation,[3] affecting the quality of data freely available online.[4]

It is critical to discern misinformation from disinformation. The key difference lies mainly in the intent of the data, i.e., if it is misleading or intended to cause harm (i.e., disinformation) or not (i.e., misinformation).[5] Consequently, fake news can be conceptualized as a form of disinformation and are strongly related to emotionally charged and sensationalized data that mimics the style of traditional news stories.[6] The differential factor of the target audience should also be considered.[7]

Therefore, provided the amount and pace of new information generation (i.e., news articles, social media posts, wiki pages), automatic data quality and validation methods have attracted the attention of the research community and the industry.[8] Such methods explore Artificial Intelligence (AI) knowledge representation schemes, Machine Learning (ML) models either for supervised or unsupervised learning, and NLP techniques.[9]

Disinformation approaches explore: (*i*) content, (*ii*) social-context (i.e., user interaction behaviors like commenting, following), or (*iii*) both.[10] Finally, deep learning-based solutions exploit neural networks.[11] Note that this research focuses on traditional ML approaches due to their

[1]Information Technologies Group, atlanTTic, University of Vigo, Vigo, Spain
[2]Research on Economics, Management and Information Technologies, Universidade Portucalense, Portugal
[3]INESC TEC, Portugal
[4]Polytechnic of Porto, Portugal

Corresponding author:
Silvia García-Méndez, Information Technologies Group, atlanTTic, University of Vigo, 36310 Spain.
Email: sgarcia@gti.uvigo.es

intrinsic interpretability compared to deep learning algorithms. In this regard, the complexity of automatic disinformation detection systems supported by advanced NLP techniques is strongly related to their opacity (i.e., complex for users to understand the results), leading to lower trustworthiness.[12]

Regarding wiki pages or articles, their collaborative nature does not come without a cost. The inability to carry out timely and exhaustive reviews of the content added by the crowd produces misleading or even low-quality articles.[13] In this regard, groups of volunteers have applied offline methods to assess the quality of Wikipedia (Available at https://en.wikipedia.org, November 2024) articles.[14] While Wikipedia has been extensively used for fact-checking, fewer studies examined their content analysis.[15–17] The aforementioned NLP and ML techniques can be exploited for that purpose.[18,19] However, in the ChatGPT (Available at https://chat.openai.com, November 2024) era, existing approaches need further development to meet the highly demanding nature of disinformation, particularly regarding interpretability and explainability.[20]

Interpretability and explainability refer to different ways of understanding the rationale behind AI-based systems.[21,22] Initial advances in eXplainable AI (XAI) were based on algorithmic interpretability by exploiting feature relevance and local approximations.[23] However, we may conclude that they are correlated to some extent, so some end users can derive meaning from comprehending how the algorithms operate (i.e., interpretability). Still, most will not and benefit from applying *post hoc* XAI techniques. The ultimate objective of these strategies is to provide qualitative meaning by linking input information (i.e., the features) with the outcome (i.e., the prediction) to make the model accountable, reliable, and trustworthy.[24] In short, interpretable AI elaborates on how the systems make the prediction, while XAI describes the reasons behind the prediction.

More in detail, interpretability refers to the visualization of relevant model parameters (e.g., feature relevance), which requires technological background, while explainability offers component-independent descriptions of the outcome.[20,25] Given that the lack of interpretability and explainability jeopardizes user confidence in the prediction,[26,27] detecting disinformation adaptively in real time is necessary.[28] Moreover, shortly, explainability will be mandatory in the European Union (Available at https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206, November 2024).

In fact, the Artificial Intelligence Act (AIA) draft in April 2021 represented a major regulatory instance on the field of AI, resulting in the final text approved in December 2023 (Available at https://data.consilium.europa.eu/doc/document/ST-5662-2024-INIT/en/pdf, November 2024). It categorizes AI in different levels of risks with different application limits, transparency requirements, and oversight mechanisms. Particular attention is paid to interpretability regarding the system's accuracy, capabilities, limitations, and use instructions. It also recognizes the right to receive clear explanations with a particular mention to the GDPR. Ultimately, the changes in February 2024 require the disclosure of the use of AI in human interactions.[23] Regarding the future legal requirements in the field of AI in the European Union, even though the act entered into force on 1st August 2024, there exists an adaptation period of two years with some exceptions. Of particular relevance are the governance rules and the obligations for general-purpose AI models, which will be applied in the first year. A more flexible deadline exists for regulated products with embedded AI capabilities. Additionally, provided the complexity of the new regulatory framework, the European Commission proposed the AI pact. (Available at https://digital-strategy.ec.europa.eu/en/policies/ai-pact, November 2024) to promote early compliance of future implementations.

The rest of this paper is organized as follows. Section 2 overviews the relevant competing disinformation solutions in the state of the art. Section 3 introduces the proposed solution, while Section 4 describes the experimental dataset, implementations, and set-up, along with the empirical evaluation results. Finally, Section 5 concludes and highlights the achievements and future work.

2 Related work

Although disinformation is a recurring problem, recent advances in the field of NLP (i.e., LLMs such as ChatGPT) have exacerbated the already high volume of publicly available misleading content, as well as its rate of propagation.[29] Consequently, due to the cost of real-time content patrolling, there is a considerable amount of unreviewed content on the web, which may contain misleading data.[14] However, few methods in the literature are dedicated to monitoring the data quality of today's most popular information sources, i.e., wiki pages.[30]

Traditional fact-checking relies on manual verification (e.g., FactCheck (Available at https://www.factcheck.org, November 2024), PolitiFact.com (Available at https://www.politifact.com, November 2024) or Snopes.com (Available at https://www.snopes.com, November 2024)). Their main drawback is scalability, given human evaluation's operational and time limitations.[31] Conversely, ML-based disinformation detectors infer content quality from text features (e.g., text length, lexical aspects such as vocabulary usage), exploit graph representations of the editorial review process (i.e., articles and editors are nodes while the corrections are edges) Bassani and Viviani,[14] or employ deep neural network models, sacrificing interpretability in favor of classification accuracy.[4] The latter prevents their practical use and explains the limited number of proposals in academic research that exploit neural strategies.[16,32]

Unfortunately, most of the existing disinformation detectors are *black boxes*, lacking the ability to explain the prediction outcome.[33] Thus, there is a need for advanced solutions that address this concern both effectively and efficiently, i.e., in real-time operation and in a way that is comprehensible not only to experts but also to end users.[34]

The explainability techniques found in the fact-checking literature are based on salience and logic. Saliency-based methods exploit the attention mechanism to highlight relevant information for end users.[35,36] More in detail, they are also called visual XAI and represent the first attribution approach applied to convolutional networks, for example, for computer vision. Its name comes after detecting, locating, and segmenting salient data by computing the feature-wise importance score.[37] Attention-based intelligibility is highly popular with *black box* disinformation detectors. They highlight tokens from the content[35] or extract relevant *n*-grams using self-attention.[38] The attention-based descriptions analyze the attention weights of neural models to show which inputs are most relevant for the model's prediction. In this strategy, apart from attention weights computation, another critical task lies in generating context vectors. Logic-based solutions describe the outcome using graphs[39] or rule mining.[40] Specifically, rule mining methods allow modeling data phenomena by exploiting numeric association rules such as the Association Rule Mining (ARM) technique, which is a condition statement.[41] While attention-based methods are the most popular technique for explaining profound learning predictions, they require advanced knowledge to maximize the information provided.[35] Moreover, they may increase the model complexity and limit its adaptability and generalization.[42] In contrast, the rule mining methods may result in highly transparent descriptions but with low readability properties.[11] A relevant advantage is that rule-based explanations are more straightforward to comprehend.[34] However, these systems are limited by the coverage of the knowledge base used for fact-checking.[20] In short, current explainability techniques (e.g., attention scores, Local Interpretable Model-agnostic Explanations - LIME, and saliency heatmaps) are unsuited for the general public, who do not have the specialized knowledge to interpret their results.[4] Finally, interpretable methods (i.e., with non *black box* properties) are considered the most straightforward approach toward direct interpretability and explainability.[43]

Focusing on the works that used wiki pages as fact-checking data sources, we must mention the solution by Sathe et al.[44] They created the WikiFactCheck-English (Available at http://github.com/WikiFactCheck-English, November 2024) dataset composed of almost 125000 entries from English Wikipedia articles. The authors validated the usefulness of this corpus with basic ML experiences. Moreover, Fu et al.[33] proposed DISCO, an explainable disinformation detection system based on a graph neural network model trained with Wikipedia articles. More recently, Brand et al.[11] proposed E-BART to detect and explain disinformation, using Wikipedia as a knowledge base.

Few solutions address the analysis of wiki articles in the literature, the works by Bassani et al.,[14] Furuta et al.[30] and Hsu et al.[16] are representative examples. Bassani et al.[14] proposed an offline supervised ML approach to address Wikipedia article quality. The authors performed experimental tests with multiple classifiers, obtaining the best results with a Gradient Boosting model. Furuta et al.[30] designed an offline fact-checking system based on a Bidirectional Encoder Representations from Transformers (BERT) model to identify the parts of articles for the editors to revise. The dataset of sentences usually revised in Wikipedia does not differentiate typographical and grammatical errors from misleading content. Consequently, this research does not address disinformation detection. Conversely, Hsu et al.[16] presented the offline Pairwise Contradiction Neural Network (PCNN) model based on the Multi-layer Perceptron (MLP) classifier to identify self-contradictory articles in Wikipedia, relying exclusively on content data. More recently, Chen et al.[45] proposed EvidenceNet based on the RoBERTa model to retrieve evidence claims from Wikipedia and predict if a statement is supported, refuted by the evidence or cannot be verified. Consequently, the system is exclusively based on content-derived data. Similarly, Petroni et al.[32] presented SIDE, a neural-network-based solution to detect untrustworthy citations on Wikipedia. Particularly, the verification scores are computed using a fine-tuned BERT model. Moreover, Shivans et al.[46] presented XFactVer, which exploits BERT for semantic similarity analysis to perform automatic cross-lingual fact-checking in Wikipedia. The authors focused on verifying references as in the work by Petroni et al.[32] Ultimately, Das et al.[47] presented a computational framework based on regular expressions to assess the quality of Wikipedia content. The authors exploited content-agnostic features (e.g., page length, number of references). Moreover, the experimental labeled data resulted from mapping the quality classes used in Wikipedia. Note that all these solutions operate offline and do not provide explainability.

2.1 Research contribution

Table 1 compares the current proposal with the most related research, considering profiling, processing, classification, and outcome explanation. As it can be seen, the prominent approach relies on transformer models (e.g., BERT), which justifies the few works that exploit dada beyond content features. It is the case of the works by Bassani et al.[14] and Das et al.[47] In light of this comparison, the current proposal is the first to explore content and side features to classify wiki pages in full streaming mode, including the relationship between users and wiki articles. Specifically,

Table 1. Comparison with related work from the literature.

Authorship	Profiling	Processing	Model	Explainability
Bassani et al.[14]	Content Review Network	Offline	Supervised ML	✗
Furuta et al.[30]	Content	Offline	BERT	✗
Hsu et al.[16]	Content	Offline	MLP	✗
Chen et al.[45]	Content	Offline	RoBERTa	✗
Petroni et al.[32]	Content	Offline	BERT	✗
Shivans et al.[46]	Content	Offline	BERT	✗
Das et al.[47]	Side	Offline	Regular expressions	✗
Proposal	Content Side	Online	Online	Textual Visual

the stream operation also applies to the word *n*-gram analysis and hyperparameter selection. The experimental results, obtained with two datasets, analyze the performance of the system in three evaluation scenarios using macro and micro metrics (i.e., accuracy, precision, recall, *F*-measure, and processing time). Finally, textual (i.e., natural language) and visual explainability are provided in the dashboard.

In contrast to previous approaches,[48] this paper expands on disinformation detection from a multidimensional perspective, i.e., considering performance, efficiency, and fairness. Moreover, it incorporates inputs from expert knowledge, also known as human-in-the-loop AI (HAI),[49] and an LLM to enhance performance and promote the explainability and fairness of the results. Both inputs are integrated into the explainability control panel: the expert knowledge is used to refine the supervised classifier model via reinforcement learning and the LLM descriptions to explain classifier predictions via highly coherent human-like generated texts. Ultimately, LLMs and the human-in-the-loop approaches can report relevant advantages to disinformation recognition, especially regarding pattern extraction and explainability.[50] We believe its combination has the potential to share the future detection methods provided the understanding capabilities of LLMs. However, these models may incorporate underlying misconceptions due to the absence of a comprehensive understanding of the problem and expert data. The human-in-the-loop approach can do it by leveraging expert knowledge of linguistic patterns, language usage, and deep semantic meaning in disinformation content. This approach also helps to limit the hallucination problem of LLMs. Consequently, our system can assist editors by reducing their efforts and time regarding disinformation detection.

In summary, the main contributions of our proposal are:

- Application of streaming techniques for identifying disinformation in wiki data. Accordingly, our solution is updated in each incoming sample, avoiding the costly batch processing that is the prominent approach in the literature.
- Generation of a wide range of features to model the evolving nature of disinformation and the crowdsourcing scheme, including the analysis of historical data.
- Use of LLM in combination with NLP techniques such as prompt engineering to generate a natural language description of the system rationale.
- The possibility of integrating an expert-in-the-loop to verify and correct the system outputs to ensure the proposal's trustworthiness, reliability, and accountability.

3 Methodology

This paper proposes an explainable solution to identify disinformation within reviews through stream-based classification. Our system focuses on wiki posts regardless of their topic (e.g., places, products, services). Note that the heterogeneity of the content is modeled with general and dataset-specific features (see Section 3.1.1).

Specifically, we seek to solve a binary classification problem and predict if a post is related to disinformation or non-disinformation content on a streaming basis. Figure 1 shows the proposed scheme composed of several modules that will be described in the subsequent sections: (*i*) stream-based data processing to gather side and content-dependent features, including those based on historical values and their analysis and selection; (*ii*) stream-based classification that evaluates different streaming scenarios (sequential, delayed and batch); and (*iii*) stream-based explainability to provide a natural language description of the model prediction based on an LLM.

Note that this section describes the decisions and techniques applied to our approach, while the software, libraries, and configuration parameters are detailed in Section 4.

3.1 Stream-based data processing

Appropriate data processing methods enable filtering non-relevant data for the ML models to perform effectively and efficiently. Data processing in streaming means that the

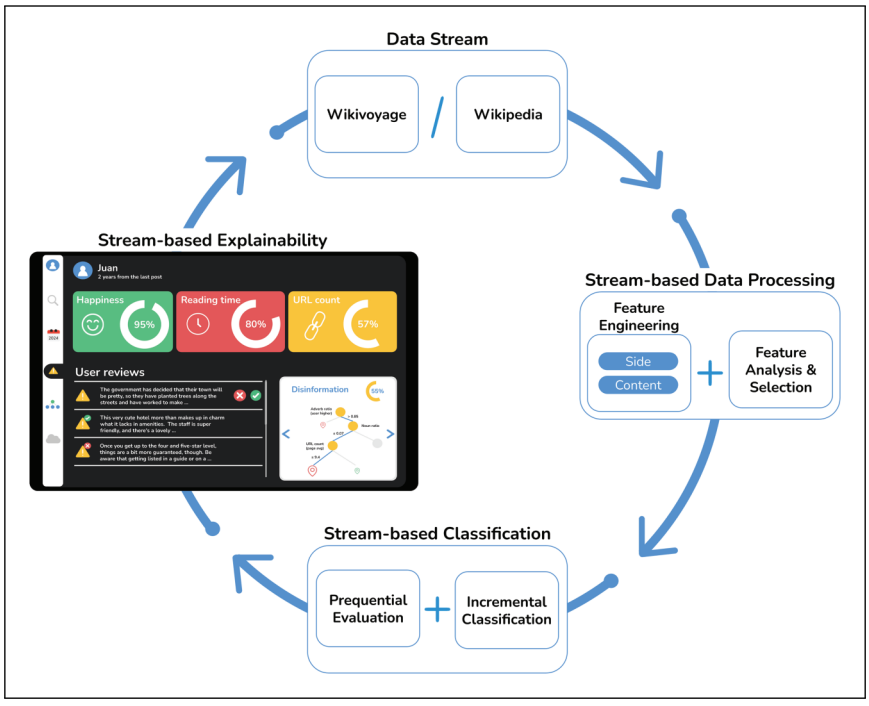

Figure 1. Stream-based classification and explanation of disinformation in wiki streams.

data stream is simulated with individual incoming samples ordered by their timestamp. In this regard, the textual content of the review is preprocessed to remove special characters, numbers, punctuation marks, and subsequent blank spaces. Stop-words are also deleted.

This stage is composed of (*i*) feature engineering to take the most advantage of the experimental data and (*ii*) feature analysis and selection to ensure that incoming data for the latter models is consistent. Note that the contributions of our work, as summarized in Section 2.1, do not rely on the existing techniques for feature engineering, analysis, and selection but on the way they are integrated into an explainable and online disinformation detection system for wiki data. In this regard, the engineered features result from an in-depth analysis of the crowdsourcing scheme and represent a relevant contribution. Special mention deserved the treatment of historical data.

3.1.1 Feature engineering. Table 2 shows the features engineered for the subsequent supervised learning stage with the ML models. While features 1-9 are generated using NLP techniques, features 10-19 are extracted directly from the wiki. Specifically, the created side features include (*i*) number of characters, words, complex words, and URL instances (Once counted, these elements are removed from the content), (*ii*) part-of-speech (POS) ratios (i.e., adjectives, adverbs, interjections, nouns, pronouns, punctuation marks, and verbs), and (*iii*) reading time and readability scores. The new content-related features are built after lemmatization. While lemmatization consists of obtaining a word's original term, considering its meaning (i.e., undoing gender, number), stemming consists of extracting the root term by removing the existing affixes directly. Consequently, the resulting root term may not exist in the language vocabulary and lack contextual meaning. We decided to perform lemmatization instead of stemming, provided its superior performance in the literature regarding natural language mining.[51] These comprise (*i*) emotion and polarity load, (*ii*) word *n*-grams obtained through a dedicated real-time *ad hoc* word *n*-gram extractor, and (*iii*) word embedding representations based on word2vect. More in detail, the *ad hoc* word *n*-gram extractor generates a *n*-size matrix in which every element represents the term and its frequency.

The posts are made by a user on a specific page based on the content topic. This behavior may be repeated if the wiki identifies a user as a bot or has reverted comments. Likewise, a page that concentrates on numerous reverts is likely arguable. For this reason, the incremental historical values grouped by user and page are calculated for each feature in Table 2. In this way, our approach is composed of (*i*) raw, (*ii*) engineered, and (*iii*) historical data (see Table 3).

3.1.2 Feature analysis & selection. The feature selection stage becomes challenging in the stream-based scenario since incoming samples may alter the classification variables. The analysis and subsequent selection of the most relevant features (original and engineered) ensure that solely valuable data are submitted to the classification model.[52] Features are filtered by a variance thresholding method (i.e., those with low variance are removed). Consequently, this

Table 2. Features explored per experimental dataset.

Source	Type	ID	Name	Description
Engineering	Side	1	Review count	Number of characters, words, difficult words, and URL instances.
		2	POS ratio	Ratio of adjectives, adverbs, interjections, nouns, pronouns, punctuation marks, and verbs.
		3	Reading time	Content reading time.
		4	Flesch score	Readability text score.
		5	McAlpine EFLAW score	Readability text score for non-native English speakers.
	Content	6	Emotion load	Anger, fear, happiness, sadness, and surprise load.
		7	Polarity	Negative, neutral, and positive load.
		8	Word2vect	Embedding representation of the content using Word2vect.
		9	Word *n*-grams	Single word-grams.
Wikivoyage	Side	10	Bot flag	It indicates if the author is a bot.
		11	Deleted flag	It indicates if the content is hidden.
		12	New flag	It indicates that it is the first revision.
		13	Revert flag	It indicates if the revision was reverted.
		14	Size difference	Difference in the number of characters added and deleted.
		15	Article quality	OK, WP10B, WP10C, WP10FA, WP10GA, WP10START, WP10STUB probability.
		16	Edit quality	False/true damaging & good faith probability.
		17	Review quality	A, B, C, D, E probability.
Wikipedia	Side	18	Article quality	OK, WP10B, WP10C, WP10FA, WP10GA, WP10START, WP10STUB probability.
		19	Edit quality	False/true damaging & good faith probability.

Table 3. Historical engineered features.

ID	Name	Description
20	User post count	Cumulative number of posts per user.
21	User spam tendency	Cumulative ratio of spam posts per user.
22	User posting antiquity	Posting antiquity per user (in weeks).
23	User posting frequency	Weekly posting frequency per user.
24, 61	User features	Incremental average and maximum per user regarding features 1 to 19 in Table 2.
62, 99	Page features	Incremental average and maximum per page regarding features 1 to 19 in Table 2.

stage aims to reduce the number of features involved in the model training, keeping only those that provide relevant knowledge. Using a dataset partition, the system computes the variance threshold per data stream in the cold start stage, which remains steady when this stage is terminated. The latter directly impacts the system's performance regarding evaluation metrics and time consumed by removing noisy data.

3.2 Stream-based classification

Figure 2 shows the pipeline of the supervised learning process performed on a stream basis or batch. In online learning, the first step involves selecting the most relevant features. Moreover, the feature selector is updated with every incoming sample. Then, the resulting category is predicted, and the model is updated with information related to that sample. Conversely, the model can be either trained or used for prediction in batch processing. The training process is slightly different from the online processing. In this case, the selector is trained, and the features are extracted. Ultimately, the training process is performed with the whole dataset, not incrementally as in online processing, preventing a continuous model update. When it comes to batch prediction, the features are first selected.

Our approximation analyzes 3 types of ML classifiers based on Gaussian probabilities, hyperplane, and trees. Note that the latter tree-based models are intrinsically interpretable and were selected based on their promising performance in similar classification problems in the literature and our experimental analyses.[53–56]

- **Gaussian Naive Bayes** (GNB) Tieppo et al.[57] implements a stream-based Naive Bayes algorithm based on Gaussian probabilities. It is used for base reference.
- **Approximate Large Margin Algorithm** (ALMA) Kang et al.[58] computes the hyperplane maximal margin for linearly separable data in streaming mode.
- **Hoeffding Adaptive Tree Classifier** (HATC) Mrabet et al.[59] consists of a single-tree decision with a branch performance evaluation function for stream-based classification.

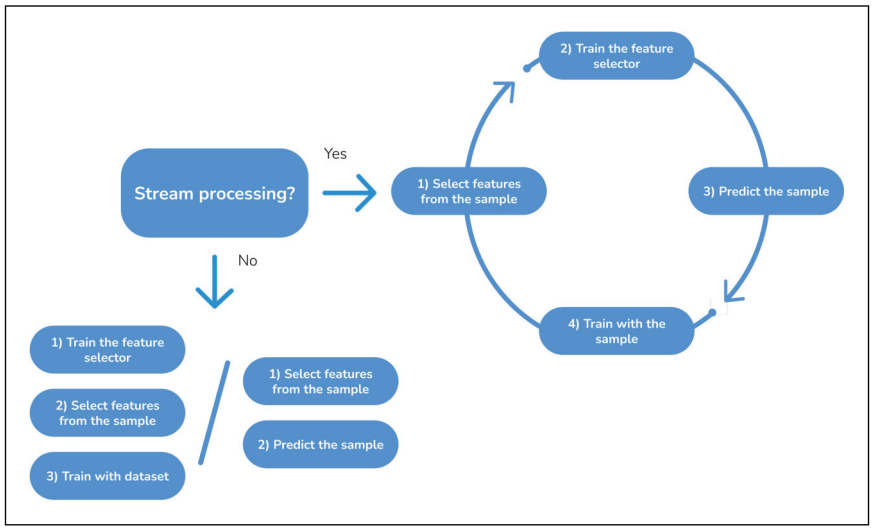

Figure 2. Streaming and batch supervised learning.

– **Adaptive Random Forest Classifier** (ARFC) Fatlawi and Kiss[60] implements an ensemble of HATC with majority voting prediction for stream mining. Specifically, we set a reduced value range for the hyperparameters to prevent a performance reduction regarding the number of estimators or instances of the model (i.e., *models* hyperparameter), features, and the number of executions per estimator (i.e., using the bagging technique and the *lambda* hyperparameter). This will ensure the model provides fast responses, i.e., in less than a second. The majority voting is computed taking into account the resulting category provided by each estimator.

As the system operates on a streaming basis, the models are updated as samples arrive one by one, ordered by their timestamp (see Figure 2). First, the features are selected from the specific sample; then, the feature selector is updated to consider the variance of the current features and the previous ones from the previous samples. In contrast to batch learning, the prediction comes first, and then the model is re-trained.

The hyperparameters are optimized at the beginning of the system execution, named the cold start stage, using a configurable set of the initial samples. In this line, the performance of the models is assessed via accuracy, precision, recall, and F-measure evaluation metrics, using macro and micro averaging. Note that micro values are especially appropriate in imbalance scenarios and to capture the presence of incorrect predictions (i.e., false positives and false negatives).[61] These metrics are updated after every new sample. This may result in an imbalance in the target classes at certain moments of the classification process. The latter is the essence of online learning compared to batch learning, which makes the classification problem more challenging. Information about the processing time is also provided.

3.3 Stream-based explainability

The explainability dashboard provides visual and natural language descriptions of the prediction outcome for algorithmic transparency and increased reliability.

Specifically, our approach generates interpretable predictions using a tree model. These models present bifurcations in each node until reaching the resulting category. Each bifurcation evaluates one of the features, hence allowing the gathering of all involved in the prediction. Then, the description in natural language is generated using an LLM exploiting prompt engineering techniques.

This process is repeated for each of the trees of a Random Forest (RF) classifier, generating as many explanations as the number of trees in the model. Note that our system removes those trees that do not predict the majority category. In addition, our system uses the history of features 1-19 to obtain the quartile in which the user's post is located, along with the three most relevant features detected by the variance selector to display them on the dashboard. Finally, it allows the evaluation of the predictions by an expert, known as the expert-in-the-loop strategy. The samples are tagged and incorporated into the streaming system for model training.

For the natural language descriptions obtained through an LLM, we exploited prompt engineering and used the post content as input (see Section 4.4 for technical details).

4 Experimental results

The experiments were executed on a computer with the following specifications:

– **Operating System**: Ubuntu 18.04.2 LTS 64 bits
– **Processor**: IntelCore i9-10900K 2.80GHz

– **RAM**: 96GB DDR4
– **Disk**: 480GB NVME + 500GB SSD

In our stream-based classification problem, the prediction is performed in the order imposed by the timestamp. Considering that the experimental data are unbalanced, different scenarios were designed to assess the system's performance in streaming mode comprehensively.

Scenario 1. The first s samples are considered for each class, where s is the number of entries for the less frequent class. In this configuration, the dataset is unbalanced by time slots. This means that the samples of a class arrive all together and are not uniformly distributed over time.
Scenario 2. The first s samples of the less represented class are considered to respect their distribution in the experimental data. Then, the s samples of the most represented class are randomly distributed to complete $2s$ entries. In this case, classes are balanced so that it is probable to find the same number of samples from both categories in a short period. For this scenario, we use the `RandomUnderSampler` library (Available at https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.RandomUnderSampler.html, November 2024).
Scenario 3. In order to evaluate the system performance in a more realistic layout than predicted with one sample and train with that sample, the training stage is delayed until the n-th sample enters the pipeline. Consequently, the data from scenario 2 is used to predict, but the subsequent training stage is performed with n samples in a row. In this scenario, a fixed delay of 100 samples was established. This means the model training takes place when the 100 samples are predicted. This allows us to replicate a real scenario where the expert-in-the-loop labels in fixed-size batches weekly.

Figure 3 details the data preparation for each scenario. The superscripts indicate the sample target class, and the lowercripts indicate the timestamp.

4.1 Experimental data

Our proposal analyzes a binary classification problem (disinformation versus non-disinformation classes) in wiki data streams. Two experimental datasets were used to analyze the solution's performance comprehensively. The creation of the datasets relies on MediaWiki (Available at https://www.mediawiki.org/wiki/MediaWiki, November 2024) which is an open-source software for wikis initially used on Wikipedia. Therefore, the gathered data was obtained via the MediaWiki Application Programming Interface (API), creating a wiki-based dataset from two wiki-based platforms: Wikivoyage and Wikipedia.

Table 4. Disinformation and non-disinformation distribution of entries per experimental dataset.

Dataset	Class	Entries
Wikivoyage	Disinformation	62186
	Non-disinformation	223576
	Total	285762
Wikipedia	Disinformation	17837
	Non-disinformation	22535
	Total	40372

Wikivoyage dataset contains contributions to travel wikis between August 2003 and June 2020. It is composed of 285762 entries, distributed between 62186 and 223576 samples of disinformation and non-disinformation entries, respectively (see Table 4). **Wikipedia** dataset contains contributions to the Wikipedia platform between September 2006 and December 2023. It is composed of 40372 entries, 17837 and 22535 for disinformation and non-disinformation, respectively (see Table 4).

4.2 Stream-based data processing

This section reports the techniques used for feature engineering, analysis, and selection and the results obtained. Regarding text processing, special characters, numbers, and punctuation marks (For numbers and punctuation marks, the removal is performed after engineering side features and before engineering the content ones. Available at https://bit.ly/41dupzh, November 2024), and stop words (Available at https://gist.github.com/sebleier/554280, November 2024) were removed using regular expressions.

4.2.1 Feature engineering. Common classification features encompass side-based (e.g., adjective ratio, reading time) and content-derived data (i.e., embedding representation and single word n-grams) as illustrated in Table 2. The features engineered per dataset are also indicated. Moreover, Table 3 reports the relational features per user and page composed from those listed in Table 2. Equation (1) represents how the accumulated historical values (for user and page) are calculated, n is the number of sessions and $X[n]$ is the historical feature evaluated in the last n sessions. Expression (a) applies to features 20 and 22, while (b) to

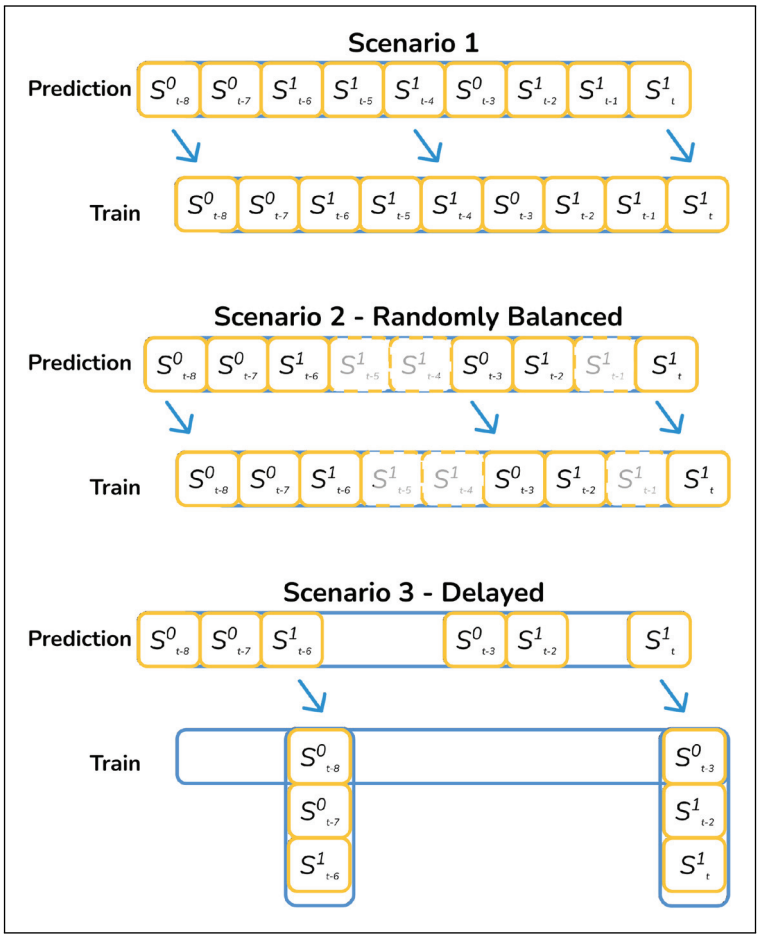

Figure 3. Data preparation for prediction and training for the three evaluation scenarios considered.

21, 23-99, and (c) to 24-99.

$$
\begin{aligned}
&\forall n \in \{1 \cdots \infty\} \\
&X[n] = \{x[0], \ldots, x[n]\} \\
&Y[n] = \{y_0[n], y_1[n], \ldots, y_{n-1}[n]\} \mid \\
&y_0[n] \leq y_1[n] \leq \ldots \leq y_{n-1}[n] \\
&\text{where; } \forall x \in X[n],\ x \in Y[n] \\
&a) sum[n] = \sum_{i=0}^{n} y_i[n] \\
&b) avg[n] = \frac{1}{n} \sum_{i=0}^{n} y_i[n] \\
&c)\ \max[n] = \max_{0 \leq i \leq n} y_i[n]
\end{aligned} \tag{1}
$$

Particularly, count features (1 in Table 2) referring to chars and words are computed using the `len` function in Python. For the latter, the textual content was firstly separated by spaces. For the difficult word count, the `textstat difficult_words` function (Available at https://pypi.org/project/textstat, November 2024) was used.

Regarding URL instances, a regular expression was implemented (Available at https://bit.ly/3N4GNM3, November 2024). The ratio features (2 in Table 2) are computed using spaCy (Available at https://spacy.io, November 2024) through the `token.pos_` function to gather their lexical category. The functions `reading_time`, `flesch_reading_ease` and `mcalpine_eflaw` of `textstat` are used to compute reading time and readability scores (features 3, 4, and 5 in Table 2).

Emotion and polarity load (features 6 and 7 in Table 2) were calculated using the `text2emotion` (Values between 0 and 1. Available at https://pypi.org/project/text2emotion), November 2024. and `TextBlob` (Values between -1 and 1. Available at https://pypi.org/project/spacytextblob, November 2024) functions, respectively.

Content feature 8 was extracted using the `word2vect` implementation of the spaCy model `en_core_web_md` (Available at https://spacy.io/models/en, November 2024). This model transforms each word into a numerical vector of 300 values and thus establishes a proximity relationship. The vector is generated for each word at the sentence level, and its average value is calculated.

Content feature 9 is generated with an online approximation of `CountVectorizer` (Available at https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html) developed *ad hoc*. Particularly, we use the `word_tokenize` (Available at https://www.nltk.org/api/nltk.tokenize.html, November 2024), `ngrams` (Available at https://tedboy.github.io/nlps/generated/generated/nltk.ngrams.html, November 2024) and `FreqDist` (Available at https://tedboy.github.io/nlps/generated/generated/nltk.FreqDist.html, November 2024). The first extracts the word tokens, the second generates the n-grams object (in our streaming scenario, 1-grams, even though this value can be configurable), and the last allows the frequency computation. A limit is established per sentence for the later single word *n*-grams to control the number of features engineered. To establish the limit above, a cold start of 0.5% of the initial samples was used to calculate the second quartile (Q2) metric, resulting in four words, which are those with a higher frequency of appearance in the sentence.

Lemmatization was performed using spaCy with the `en_core_web_md` model prior to engineering the content features but after engineering the side ones.

In total, 89 features were engineered: 80 historical features, 4 content, and 5 side features.

4.2.2 Feature analysis & selection. The variance threshold was established based on experimental tests as the 90th percentile feature variance value using the 0.5% of the initial dataset (cold start) using probabilistic features 15-17 and features 18-19 in the Wikivoyage and Wikipedia experimental data, respectively, resulting in 0.067 and 0.022 (These values do not change dynamically). The `VarianceThreshold` method (Available at https://riverml.xyz/0.11.1/api/feature-selection/VarianceThreshold, November 2024) from the `River` package (Available at https://riverml.xyz/0.11.1, November 2024) enables feature variance computation. This process is dynamic, and the selection of features varies for each sample. In the last sample, 80 and 65 features were discarded for Wikivoyage and Wikipedia datasets, respectively. Reducing features in a streaming system is appropriate and important to avoid resource consumption and keep it at a moderate margin, promoting scalability. Note that the features discarded are those with low variance, so their impact on the model's performance is not expected to be significant. Moreover, removing irrelevant features prevents overfitting in the tree-based classifiers.[62]

4.3 Stream-based classification

Stream-based ML models were incrementally trained and evaluated using an *ad hoc* implementation of `EvaluatePrequential` (Available at https://scikit-multiflow.readthedocs.io/en/stable/api/generated/skmultiflow.evaluation.EvaluatePrequential.html, November 2024).

Table 5. Hyper-parameter configuration (best values in bold).

Classifier	Hyperparameter
ALMA	alpha = [0.3, 0.5, 0.7, **0.9**]
	B = [0.6, 1.0, 1.4, **1.8**]
	C = [0.6, 1.0, 1.1, 1.4, **1.8**]
HATC	depth = [None, 50, 100, **200**]
	tiethreshold = [0.9, 0.5, 0.05, **0.005**]
	maxsize = [25, 50, 100, **200**]
ARFC	models = [10, 25, 50, **75**]
	features = [sqrt, 25, 50, **100**]
	lambda = [5, 25, 50, **100**]

The implementations of the ML models come from the River package (Due to computational performance, training, and testing were updated every ten slots): GNB (Available at https://riverml.xyz/dev/api/naive-bayes/GaussianNB, November 2024), ALMA (Available at https://riverml.xyz/0.11.1/api/linear-model/ALMAClassifier, November 2024), HATC (Available at https://riverml.xyz/0.11.1/api/tree/HoeffdingAdaptiveTreeClassifier, November 2024) and ARFC (Available at https://riverml.xyz/0.11.1/api/ensemble/AdaptiveRandomForestClassifier, November 2024). Table 5 shows the hyperparameter optimization ranges used from which the following optimal values were selected for both datasets. Hyperparameter optimization is performed with an exhaustive search over the hyperparameters ranges similar to the operation of the `GridSearchCV` library (Available at https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html, November 2024) in batch with the 0.5% of the original samples as cold start. Moreover, in our binary classification problem, we use 0 and 1 to refer to the non-disinformation and disinformation classes, respectively.

Table 6 shows the evaluation result for each scenario in the Wikivoyage dataset. The results in scenario 1 are above 81%, 78%, 65%, and 72% for the GNB, ALMA, HATC, and ARFC models, respectively. As the problem becomes more challenging and realistic in scenarios 2 and 3, the performance of the models changes significantly. Particularly, for scenario 2, the results are above 66%, 52%, 74%, and 83%. A similar performance is obtained in scenario 3. More in detail, the performance of GNB drops (e.g., 0.657 for the recall of the disinformation class in scenario 2). A similar phenomenon is exhibited by the ALMA model (e.g., 0.481 in the case of the non-disinformation recall in scenario 3). These two models, GNB and ALMA are the fastest. Moreover, the HATC algorithm attains better results than GNB and ALMA, and the processing time is in the same magnitude order. However, the values are still variable for this model (e.g., 0.651 to 1.00 between the disinformation class versus the non-disinformation class for the recall

Table 6. Classification results for Wikivoyage dataset (#0: non-disinformation, #1: disinformation).

			Precision			Recall			*F*-measure			Time
	Model	Acc.	Macro	#0	#1	Macro	#0	#1	Macro	#0	#1	(s)
1	GNB	82.00	0.820	**0.825**	0.815	0.820	0.814	**0.826**	0.820	0.819	0.821	20.00
	ALMA	77.79	0.778	0.779	0.777	0.778	0.779	0.777	0.778	0.779	0.777	10.07
	HATC	82.58	0.871	0.742	**1.000**	0.825	**1.000**	0.651	0.820	0.852	0.788	40.88
	ARFC	**85.74**	**0.886**	0.781	0.991	**0.857**	0.994	0.721	**0.855**	**0.875**	**0.834**	1379.36
2	GNB	74.32	0.750	0.711	0.789	0.742	0.828	0.657	0.741	0.765	0.717	24.26
	ALMA	52.19	0.522	0.527	0.517	0.522	0.527	0.517	0.522	0.527	0.517	12.38
	HATC	77.91	0.780	0.764	0.797	0.779	0.815	0.743	0.779	0.789	0.769	64.15
	ARFC	**88.04**	**0.884**	**0.927**	**0.842**	**0.881**	**0.829**	**0.933**	**0.880**	**0.875**	**0.885**	2701.03
3	GNB	76.76	0.769	0.750	0.789	0.767	0.810	0.724	0.767	0.779	0.755	23.83
	ALMA	50.64	0.507	0.513	0.501	0.507	0.481	0.532	0.506	0.496	0.516	12.33
	HATC	80.87	0.809	0.824	0.794	0.809	0.791	0.827	0.809	0.807	0.811	63.65
	ARFC	**87.67**	**0.881**	**0.925**	**0.837**	**0.877**	**0.822**	**0.932**	**0.876**	**0.871**	**0.882**	2749.25

Table 7. Classification results for wikipedia dataset (#0: non-disinformation, #1: disinformation).

			Precision			Recall			*F*-measure			Time
	Model	Acc.	Macro	#0	#1	Macro	#0	#1	Macro	#0	#1	(s)
1	GNB	85.53	0.855	0.854	0.857	0.855	0.855	0.856	0.855	0.855	0.856	6.28
	ALMA	81.70	0.817	0.816	0.818	0.817	0.816	0.818	0.817	0.816	0.818	2.94
	HATC	83.49	0.835	0.822	0.848	0.835	0.852	0.818	0.835	0.837	0.833	13.29
	ARFC	**92.40**	**0.924**	**0.925**	**0.923**	**0.924**	**0.922**	**0.926**	**0.924**	**0.924**	**0.925**	503.40
2	GNB	83.15	0.833	0.857	0.809	0.832	0.800	0.864	0.831	0.827	0.836	6.13
	ALMA	72.59	0.726	0.728	0.724	0.726	0.728	0.724	0.726	0.728	0.724	3.90
	HATC	81.05	0.811	0.817	0.805	0.811	0.805	0.816	0.811	0.811	0.810	12.89
	ARFC	**90.33**	**0.904**	**0.923**	**0.885**	**0.904**	**0.882**	**0.925**	**0.903**	**0.902**	**0.905**	615.00
3	GNB	80.82	0.809	0.831	0.788	0.809	0.778	0.839	0.808	0.804	0.813	6.11
	ALMA	63.34	0.634	0.642	0.626	0.634	0.618	0.649	0.633	0.630	0.637	2.99
	HATC	72.81	0.728	0.735	0.722	0.728	0.721	0.735	0.728	0.728	0.728	13.21
	ARFC	**88.90**	**0.889**	**0.902**	**0.877**	**0.889**	**0.875**	**0.903**	**0.889**	**0.888**	**0.890**	574.56

metric in scenario 1. Ultimately, the ARFC model exhibits a more stable performance even though it is more time-consuming (i.e., values above 80% in all evaluation metrics of scenario 2 and 3). Note that this model can process up to 45 samples per second (62186 non-disinformation and 62186 disinformation samples in 2749.25s) (0.02 seconds per sample) in scenario 3. This ensures its operation in real time. Moreover, the latter two models, HATC and ARFC, are interpretable.

Table 7 shows the results using the Wikipedia dataset. In this case, the dataset is smaller but quite balanced, allowing us to see each scenario's effect more clearly. Although the behavior is very similar to the previous dataset, it presents a few drops in performance in scenario 3, being ALMA the most affected classifier. ARFC offers stable behavior, with variations of approximately 2% concerning scenario 2. Furthermore, ARFC has values above 85% in all its metrics and scenarios, being especially favorable in scenario 1, with values above 90%. It maintains time performance in scenario 3 with the capacity to process 62 samples per second (17837 non-disinformation and 17837 disinformation samples in 574.56s) and one sample in 0.02 seconds.

Compared to the results obtained for the Wikivoyage dataset, the ones displayed in Table 7 are more stable regardless of classifiers and scenarios. Based on the discussion provided for Table 6, ARFC continues to be the best-performing model. Elaborating on scenario 3, the most challenging experiment, the values are close to 80%, above 60%, above 70%, and close to 90% for GNB, ALMA, HATC, and ARFC models.

Classifiers based on Gaussian probabilities (e.g., GNB) and hyperplanes (e.g., ALMA) are more robust to unbalanced data compared to tree-based classifiers (e.g., HATC and ARFC).[63] The reason is that each incoming sample results in the decision path's reconfiguration since it is entirely dependent on the input features. Consequently, HATC and ARFC exhibit more significant differences in the recall metric for scenario 1 in the bigger dataset (i.e., Wikivoyage) between the disinformation and non-disinformation classes than GNB and ALMA.

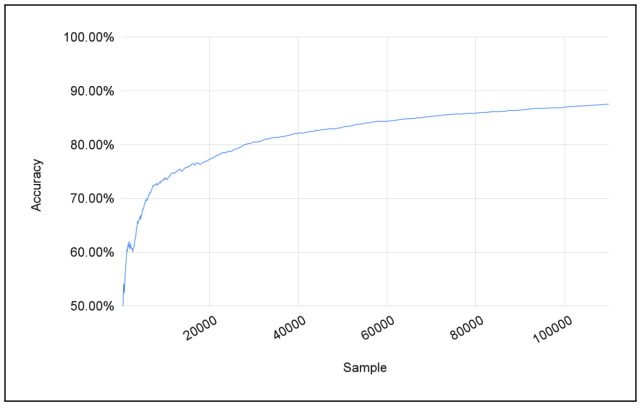

Figure 4. Accuracy evolution in Wikivoyage dataset.

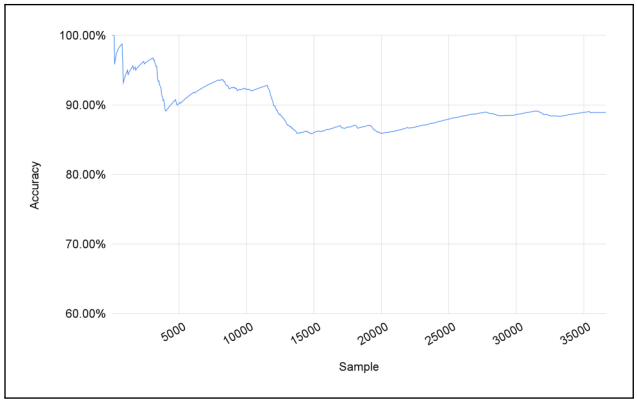

Figure 5. Accuracy evolution in Wikipedia dataset.

Figures 4 and 5 display the evolution of accuracy for scenario 3 for Wikivoyage and Wikipedia datasets, respectively. In the first phases, both datasets present their main prediction errors that stabilize and minimize as the sample grows. Due to the delay in training in this scenario, tiny abrupt drops are observed until the model is trained. Especially in the Wikipedia dataset, the curve presents more significant fluctuations due to the smaller number of samples.

Elaborating on the error analysis for the ARFC model in scenario 1, a discussion on the characteristics of the disinformation posts that are more challenging, provided the evolving nature of the system, is appropriate. In this regard and focusing on the Wikivoyage dataset, the incorrectly predicted entries exhibit higher values for feature WP10STUB. Similarly, in the Wikipedia dataset, many difficult words and a high value for WP10FA show a detrimental effect. This behavior aligns with the ORES article quality probability scheme in which STUB refers to concise posts and FA to those who receive a positive score by the reviewers (Available at https://en.wikipedia.org/wiki/Wikipedia:Content_assessment, November 2024). Consequently, it is reasonable that the system fails when little information is provided, the data are complex (a characteristic of disinformation content[64]), or the post scored high in manual verification.

4.4 Stream-based explainability

The explainability dashboard is presented in Figure 6. At the top, information related to the username and the last contribution are displayed. The three most relevant features used for prediction are shown below. The green, yellow, and red colors indicate whether the value of the feature has exceeded quartiles 1, 2, and 3, respectively, concerning all users. The user's contributions are shown at the bottom left. The human-in-the-loop can correct the system by validating or rejecting the predictions in this section. The evaluation option is only available for those not previously corrected or validated. In the bottom right part, the route of a decision tree and the selected features are shown until reaching the predicted category. Note that this decision path visualization applies only to tree-based models (HATC or ARFC) in our experiments. The end user can select the trees that result in the majority label, and the natural language description is updated accordingly (see Figure 7). The prediction confidence percentage based on `Predict_Proba_One` function (Available at https://riverml.xyz/0.11.1/api/base/Classifier, November 2024) is also provided.

Particularly, for the natural language explanation in Figure 7, we exploited the GPT3.5-turbo (Available at https://openai.com/product, November 2024) model using the default parameters, the following prompt, and the post content. An example of the output provided by the LLM is shown in Figure 7.

- Prompt: *Our Machine Learning model has predicted that this text <Text> is classified as <Category> with a confidence of <Confidence>%. The most relevant path features are: [List of relevant features]. Generate a human-explainable text that summarizes the decision made by the classifier.*

4.5 Discussion

The proposed method addresses the problem of identifying and explaining disinformation in wiki platforms using wiki events as streams. However, since the surveyed works use an offline processing approach, comparing the results may not be straightforward (see Table 1).

The Wikipedia article quality assessment solution (i.e., to distinguish those with high or low-quality content) by Bassani et al.[14] attained similar results as our proposal (91.40% accuracy with 0.085 Mean Squared Error - MSE) with the Gradient Boosting model (see Table 8). Moreover, Furuta et al.[30] proposed a fact-checking assistant for Wikipedia. However, the accuracy obtained with the BERT model is low (69.10%, 23.3 percent points lower than the ARFC model in scenario 1 for the Wikipedia dataset). Hsu et al.[16] created WikiContradiction to detect

Figure 6. Explainability dashboard.

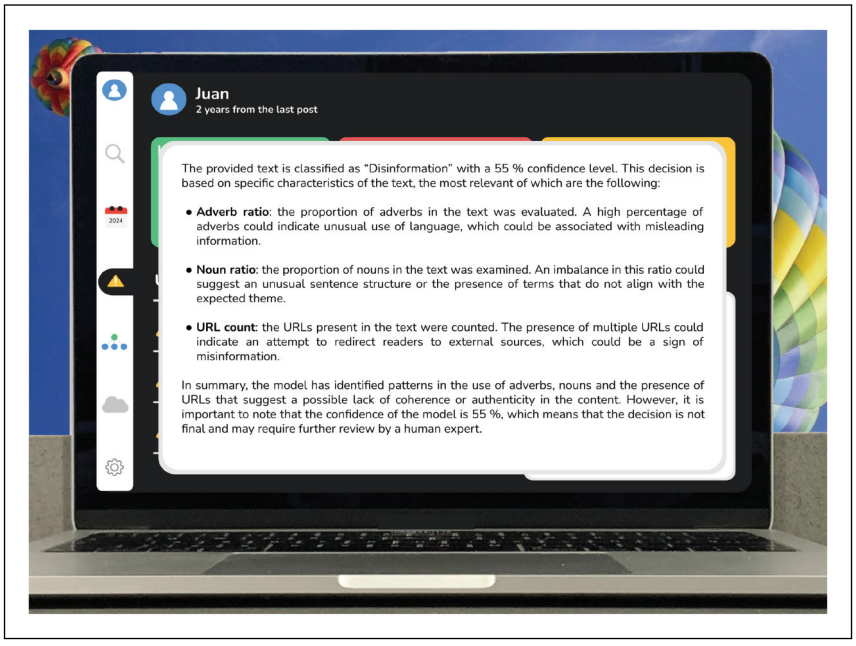

Figure 7. Pop-up explainability of the decision tree path.

self-contraction articles for Wikipedia. The best result obtained with a balanced setting and test-train ratio of 80% with the MLP model is 63% for accuracy (27 percent points lower than the ARFC in scenario 2 for the Wikipedia dataset). Regarding the more recent works, the solution by Chen et al.[45] attained the best results with the RoBERTa model, however below the 80% threshold. Similarly, Petroni et al.[32] and Shivans et al.[46] proposed to exploit the BERT model which provided up to 85% accuracy (7 percent points lower than the ARFC model in scenario 1 for the Wikipedia dataset). Ultimately, the solution by Das et al.[47] based on regular expressions attained similar results to those by Petroni et al.[32]

Unfortunately, these works discussed do not provide training time information, operate offline, and lack explainability capabilities.

Table 8. Comparison with the best approach from the literature.

Proposal	Technique	Accuracy	Time
Furuta et al.[30]	BERT	69.10	NA
Hsu et al.[16]	MLP	63.12	NA
Chen et al.[45]	RoBERTa	76.95	NA
Petroni et al.[32]	BERT	85.69	NA
Shivans et al.[46]	BERT	70.52	NA
Das et al.[47]	Regular expressions	83.90	NA

5 Conclusions

Unverified sources of information (e.g., social media platforms, and wiki pages) pose a significant threat regarding trustworthiness and data accuracy, mainly impacting the quality of the information generated and disseminated among end users. Apart from low-quality content, the spread of misleading data and misinformation is particularly relevant. The current data quality and validation methods in the literature cannot carry out timely and exhaustive reviews of the content added by the crowd. Accordingly, this work focused on wiki pages, a popular data source for fact-checking analysis, even though fewer studies examined its content.

Our solution comprises stream-based data processing with feature engineering (content and side features) and feature analysis and selection, stream-based classification, and real-time explanation of the prediction outcome. The explainability dashboard was intended for the general public without requiring specialized knowledge to interpret the decision path of the classification model. Moreover, the system's performance was evaluated with two datasets in three scenarios, attaining evaluation metrics around 90%. Thus enhancing the performance of existing competing solutions in the literature. Summing up, it contributes to reducing the cost of real-time content patrolling of disinformation data in an automatic and scalable way compared to traditional approaches, which rely exclusively on manual supervision.

Future work will leverage the system's modular design to integrate claim verification through evidence retrieval. Some technical design decisions will also be re-evaluated, such as the dynamic adjustment of the variance threshold for feature analysis and selection to address potential changes in feature importance over time (i.e., concept drift), sensitivity analysis on different delays for scenario 3 and alternative techniques such as mutual information and forward selection. Multilingual processing and using an LLM for feature engineering will also be explored. The system will be made available to the research community via API to analyze user feedback, for example, regarding the explainability dashboard. Additionally, we plan to perform new experiments using the self-supervised learning strategy[65], ensemble learning,[66] and neural algorithms[67] as well as auto-ML approaches.

Statements and declarations

Funding

This work was partially supported by: (*i*) Xunta de Galicia grants ED481B-2022-093 and ED481D 2024/014, Spain; and (*ii*) Portuguese National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) as part of project UIDB/50014/2020.

Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

References

1. Caled D and Silva MJ. Digital media and misinformation: An outlook on multidisciplinary strategies against manipulation. *J Comput Soc Sci* 2022; 5: 123–159.
2. Alexiadis A, Veliskaki A, Nizamis A, et al. A smarthome conversational agent performing implicit demand-response application planning. *Integr Comput Aided Eng* 2021; 29: 43–61.
3. Marcondes FS, Almeida JJ and Novais P. An exploratory design science research on troll factories. *Integr Comput Aided Eng* 2023; 31: 95–115.
4. Gurrapu S, Huang L and Batarseh FA. ExClaim: Explainable Neural Claim Verification Using Rationalization. In: *Proceedings of the IEEE annual software technology conference*, 2022, pp.19–26. IEEE, https//doi:org/10.1109/STC55697.2022.00012.
5. Broda E and Strömbäck J. Misinformation, disinformation, and fake news: lessons from an interdisciplinary, systematic literature review. *Ann Int Commun Assoc* 2024; 48: 139–166.
6. Zrnec A, Poženel M and Lavbic D. Users' ability to perceive misinformation: An information quality assessment approach. *Inf Process Manag* 2022; 59: 102739.
7. Zhou X and Zafarani R. A survey of fake news: fundamental theories, detection methods, and opportunities. *ACM Comput Surv* 2020; 53: 1–40.
8. Bordel B, Alcarria R, Martín D, et al. An agent-based method for trust graph calculation in resource constrained environments. *Integr Comput Aided Eng* 2019; 27: 37–56.
9. Guo Z, Schlichtkrull M and Vlachos A. A survey on automated fact-checking. *Trans Assoc Comput Linguist* 2022; 10: 178–206.
10. García-Méndez S, Leal F, Malheiro B, et al. Simulation, modelling and classification of wiki contributors: spotting the good, the bad, and the ugly. *Simul Model Pract Theory* 2022b; 120: 102616.
11. Brand E, Roitero K, Soprano M, et al. A neural model to jointly predict and explain truthfulness of statements. *J Data Inf Quality* 2023; 15: 1–19.
12. Toreini E, Aitken M, Coopamootoo K, et al. The relationship between trust in AI and trustworthy machine learning technologies. In: *Proceedings of the conference on fairness, accountability, and transparency*, 2020, pp.272–283.

Association for Computing Machinery, https://doi.org/10.1145/3351095.3372834.
13. Leal F, García-Méndez S, Malheiro B, et al. Explanation Plug-In for Stream-Based Collaborative Filtering. In: *Lecture Notes in Networks and Systems*, 2022, Vol. 468, pp.42–51. https://doi.org/10.1007/978-3-031-04826-5 5.
14. Bassani E and Viviani M. Automatically assessing the quality of Wikipedia contents. In: *Proceedings of the ACM/SIGAPP symposium on applied computing*, 2019, volume Part F147772, pp.804–807. Association for Computing Machinery. https://doi.org/10.1145/3297280.3297357.
15. Chernyavskiy A, Ilvovsky D and Nakov P. WhatTheWikiFact: Fact-Checking Claims Against Wikipedia. In: *Proceedings of the 30th ACM international conference on information & knowledge management*, 2021, pp.4690–4695. Association for Computing Machinery. https://doi.org/10.1145/3459637.3481987.
16. Hsu C, Li C-T, Saez-Trumper D, et al. WikiContradiction: Detecting Self-Contradiction Articles on Wikipedia. In: *2021 IEEE international conference on big data (Big Data)*, 2021, pp.427–436. IEEE. https://doi.org/10.1109/BigData52589.2021.9671319.
17. Trokhymovych M and Saez-Trumper D. WikiCheck: An end-to-end open source Automatic Fact-Checking API based on Wikipedia. In: *Proceedings of the 30th ACM international conference on information & knowledge management*, 2021, pp.4155–4164. Association for Computing Machinery. https://doi.org/10.1145/3459637.3481961.
18. García-Méndez S, de Arriba-Pérez F, Barros-Vila A, et al. Detection of temporality at discourse level on financial news by combining natural language processing and machine learning. *Expert Syst Appl* 2022a; 197: 116648.
19. Islam S, Abba A, Ismail U, et al. Vulnerability prediction for secure healthcare supply chain service delivery. *Integr Comput Aided Eng* 2022; 29: 389–409.
20. Das A, Liu H, Kovatchev V, et al. The state of human-centered NLP technology for fact-checking. *Inf Process Manag* 2023; 60: 103219.
21. Macas B, Garrigós J, Martínez JJ, et al. An explainable machine learning system for left bundle branch block detection and classification. *Integr Comput Aided Eng* 2023; 31: 43–58.
22. Mercaldo F, Giammarco MD, Ravelli F, et al. Alzheimer's disease evaluation through visual explainability by means of convolutional neural networks. *Int J Neural Syst* 2024; 34: 2450007.
23. Nannini L, Alonso-Moral J, Catala A, et al. Operationalizing explainable AI in the EU regulatory ecosystem. *IEEE Intell Syst* 2024; 39: 37–48 https://doi.org/10.1109/MIS.2024.3383155
24. Lisboa P, Saralajew S, Vellido A, et al. The coming of age of interpretable and explainable machine learning models. *Neurocomputing* 2023; 535: 25–39.
25. Salazar-González JL, Luna-Romera JM, Carranza-García M, et al. Enhancing smart home appliance recognition with wavelet and scalogram analysis using data augmentation. *Integr Comput Aided Eng* 2024; 31: 307–326.
26. Kotonya N and Toni F. Explainable automated fact-checking: a survey. In: *Proceedings of the international conference on computational linguistics*, 2020, pp.5430–5443. International Committee on Computational Linguistics. https://doi.org/10.18653/v1/2020.colingmain.474.
27. Si J, Zhu Y and Zhou D. Exploring faithful rationale for multi-hop fact verification via salience-aware graph learning. *Proc AAAI Conf Artif Intell* 2023; 37: 13573–13581.
28. Kozik R, Kula S, Choras M, et al. Technical solution to counter potential crime: text analysis to detect fake news and disinformation. *J Comput Sci* 2022; 60: 101576.
29. Pennycook G, Epstein Z, Mosleh M, et al. Shifting attention to accuracy can reduce misinformation online. *Nature* 2021; 592: 590–595.
30. Furuta T and Suzuki Y. A Fact-checking Assistant System for Textual Documents. In: *Proceedings of the international conference on multimedia information processing and retrieval*, 2021, pp.243–246. IEEE. https://doi.org/10.1109/MIPR51284.2021.00046.
31. Sharma K, Qian F, Jiang H, et al. Combating fake news: a survey on identification and mitigation techniques. *ACM Trans Intell Syst Technol* 2019; 10: 21–62.
32. Petroni F, Broscheit S, Piktus A, et al. Improving wikipedia verifiability with AI. *Nat Mach Intell* 2023; 5: 1142–1148.
33. Fu D, Ban Y, Tong H, et al. DISCO: comprehensive and explainable disinformation detection. In: *Proceedings of the ACM international conference on information & knowledge management*, 2022, pp.4848–4852. Association for Computing Machinery. https://doi.org/10.1145/3511808.3557202.
34. Leal F, Veloso B, Malheiro B, et al. Stream-based explainable recommendations via blockchain profiling. *Integr Comput Aided Eng* 2021; 29: 105–121.
35. Shu K, Cui L, Wang S, et al. DEFEND: xplainable Fake News Detection. In: *Proceedings of the ACM SIGKDD international conference on knowledge discovery & data mining*, 2019, pp.395–405. Association for Computing Machinery. https://doi.org/10.1145/3292500.3330935.
36. Wu L, Rao Y, Liang H, et al. DTCA: decision tree-based Co-Attention networks for explainable claim verification. In: *Proceedings of the annual meeting of the association for computational linguistics*, 2020, pp.1024–1035. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.97.
37. Borys K, Schmitt YA, Nauta M, et al. Explainable AI in medical imaging: an overview for clinical practitioners – saliency-based XAI approaches. *Eur J Radiol* 2023; 162: 110787.
38. Yang F, Pentyala SK, Mohseni S, et al. XFake: Explainable fake news detector with visualizations. In: *Proceedings of the world swide web conference*, 2019, pp.3600–3604. Association for Computing Machinery. https://doi.org/10.1145/3308558.3314119.

39. Denaux R and Gomez-Perez JM. Linked credibility reviews for explainable misinformation detection. In: *Lecture Notes in Computer Science including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*, volume 12506 LNCS, 2020, pp.147–163. Springer. https://doi.org/10.1007/978-3-030-62419-4 9.
40. Ahmadi N, Lee J, Papotti P, et al. Explainable fact checking with probabilistic answer set programming. In: *Proceedings of the conference for truth and trust Online*, 2019, pp.1–9. TTO Conference Ltd. https://doi.org/10.36370/tto.2019.15.
41. Veerappa M, Anneken M and Burkart N. Evaluation of interpretable association rule mining methods on time-series in the maritime domain. In: *Lecture Notes in Computer Science including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*, volume 12663 LNCS, 2021, pp.204–218. https://doi.org/10.1007/978-3-030-68796-0 15.
42. Jin D, Sergeeva E, Weng W, et al. Explainable deep learning in healthcare: a methodological survey from an attribution view. *WIREs Mechan Disease* 2022; 14: 1–25.
43. Kotonya N, Spooner T, Magazzeni D, et al. Graph reasoning with context-aware linearization for interpretable fact extraction and verification. In: *Proceedings of the fourth workshop on fact extraction and VERification (FEVER)*, 2021, pp.21–30. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.fever-1.3.
44. Sathe A, Ather S, Le TM, et al. Automated fact-checking of claims from wikipedia. In: *Proceedings of the international conference on language resources and evaluation*, 2020, pp.6874–6882.
45. Chen Z, Hui SC, Zhuang F, et al. EvidenceNet: evidence fusion network for fact verification. In: *Proceedings of the ACM web conference*, 2022, pp.2636–2645. Association for Computing Machinery. https://doi.org/10.1145/3485447.3512135.
46. Shivansh S, Ankita M, Aakash J, et al. Cross-Lingual fact checking: automated extraction and verification of information from wikipedia using references. In: *Proceedings of the international conference on natural language processing*, 2023, pp.828–831. Association for Computational Linguistics.
47. Das P, Johnson I, Saez-Trumper D, et al. Language-Agnostic modeling of wikipedia articles for content quality assessment across languages. In: *Proceedings of the international AAAI conference on web and social media*, 2024, volume 18, pp.1924–1934. AAAI Press.
48. García-Méndez S, Leal F, de Arriba-Pérez F, et al. Explainable classification of wiki streams. In: *Proceedings of the world conference on information systems and echnologies*, 2023, Springer. To appear.
49. Demartini G, Mizzaro S and Spina D. Human-in-the-loop artificial intelligence for fighting online misinformation: challenges and opportunities. *Bull IEEE Comput Soc Techn Committ Data Eng* 2020; 43: 65–74.
50. Hanafi M, Katsis Y, Jindal I, et al. A comparative analysis between Human-in-the-loop systems and large language models for pattern extraction tasks. In: *Proceedings of the workshop on data science with Human-in-the-Loop*, 2022, pp.43–50. Association for Computational Linguistics.
51. Mounika MK, Hemalatha B, Mounka MS, et al. Speech/Text to sign language convertor using NLP. *UGC Care Group I J* 2022; 12: 17–22.
52. Zhang C, Xue Y, Neri F, et al. Multi-objective self-adaptive particle swarm optimization for large-scale feature selection in classification. *Int J Neural Syst* 2024; 34: 2450014.
53. Garg S and Sharma DK. Linguistic features based framework for automatic fake news detection. *Comput Indus Eng* 2022; 172: 108432.
54. Patil S and Dinesha H. URL redirection attack mitigation in social communication platform using data imbalance aware machine learning algorithm. *Indian J Sci Technol* 2022; 15: 481–488.
55. de Arriba-Pérez F, García-Méndez S, Leal F, et al. Exposing and explaining fake news on-the-fly. *Mach Learn* 2024; 113: 1–23.
56. Vyas P, Liu J and Xu S. Real-Time Fake News Detection on the X (Twitter): An Online Machine Learning Approach. In: *Proceedings of the Americas conference on information systems*, 2024, pp.1339.
57. Tieppo E, Barddal JP and Nievola JC. Classifying potentially unbounded hierarchical data streams with incremental Gaussian Naive Bayes. In: *Lecture notes in computer science including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*, 2021, volume 13073 LNAI, pp.421–436. Springer. https://doi.org/10.1007/978-3-030-91702-9 28.
58. Kang S, Kim D and Cho S. Approximate training of one-class support vector machines using expected margin. *Comput Indus Eng* 2019; 130: 772–778.
59. Mrabet ZE, Selvaraj DF and Ranganathan P. Adaptive hoeffding tree with transfer learning for streaming synchrophasor data sets. In: *Proceedings of the IEEE international conference on big data*, 2019, pp.5697–5704. IEEE. https://doi.org/10.1109/BigData47090.2019.9005720.
60. Fatlawi HK and Kiss A. On robustness of adaptive random forest classifier on biomedical data stream. In: *Lecture notes in computer science including subseries lecture notes in Artificial intelligence and lecture notes in bioinformatics*, 2020, volume 12033 LNAI, pp.332–344. Springer. https://doi.org/10.1007/978-3-030-41964-629.
61. Vanacore A, Pellegrino MS and Ciardiello A. Evaluating classifier predictive performance in multi-class problems with balanced and imbalanced data sets. *Qual Reliab Eng Int* 2023; 39: 651–669.

62. Oladimeji OO, Oladimeji A and Oladimeji O. Classification models for likelihood prediction of diabetes at early stage using feature selection. *Appl Comput Inf* 2024; 20: 279–286.
63. Alkharabsheh K, Alawadi S, Kebande VR, et al. A comparison of machine learning algorithms on design smell detection using balanced and imbalanced dataset: a study of god class. *Inf Softw Technol* 2022; 143: 106736.
64. Janicka M, Pszona M and Wawer A. Cross-domain failures of fake news detection. *Comput Sist* 2019; 23: 1089–1097.
65. Rafiei MH, Gauthier LV, Adeli H, et al. Self-supervised learning for electroencephalography. *IEEE Trans Neural Netw Learn Syst* 2022; 35: 1–15.
66. Alam KMR, Siddique N and Adeli H. A dynamic ensemble learning algorithm for neural networks. *Neural Comput Appl* 2020; 32: 8675–8690.
67. Rafiei MH and Adeli H. A new neural dynamic classification algorithm. *IEEE Trans Neural Netw Learn Syst* 2017; 28: 3074–3083.