UIT-News-Verification: The end to end system for News Verification*

Chau Tan[1], Nguyen Hung Trung Hieu[1], Trinh The Hien[1], and Trong-Hop Do

Faculty of Information Science and Engineering
University of Information and Technology
Ho Chi Minh City Viet Nam National University, Ho Chi Minh City, Vietnam
`20520926@gm.uit.edu.vn`, `20521342@gm.uit.edu.vn`, `20521310@gm.uit.edu.vn` and `hopdt@uit.edu.vn`

Abstract. The verification of news is a pressing concern in the today's digital age, where the spread of fake news can have far-reaching consequences. With the rise of social media, anyone can share and distribute news on a large scale, making it difficult to distinguish between real and fake news. In Vietnam, the problem of fake news is especially prevalent and complex, due to the spread of fake news and misinformation on social media platforms. This has led to heightened concerns over the impact of these false reports on public opinion and the possibility of social unrest. To address the issue of news chaotics in Vietnam, online learning techniques can be suitable. This techniques can cope with the large-scale, heterogeneous, and evolving nature of online news data. Additionally, online learning techniques can adapt to changing patterns and trends in news chaotics, ensuring that the detection system stays up-to-date. Finally, the use of online learning techniques can help to automate the detection of fake news, providing a faster and more efficient solution than manual detection. Our approach to detecting fake news in Vietnam incorporates a range of deep learning methods, including graph neural networks, convolutional neural networks (CNN), long short-term memory networks (LSTM), gated recurrent units (GRU), and support vector machines (SVM). By using these methods, we aim to automate the detection of fake news and reduce the spread of false information in Vietnam. Overall, our approach shows promise for improving the accuracy and efficiency of fake news detection in the modern media landscape.

1 Introduction

In recent years, the spread of fake news has become a significant issue, leading to the dissemination of misinformation and disinformation on a global scale. News verification problems arise when it becomes difficult to differentiate between real and fake news, and this presents a significant challenge for society. With the rise of digital media and the ease of sharing information online, it has become increasingly challenging to verify the authenticity and accuracy of news content. This is compounded by the prevalence of social media platforms, which enable the rapid spread of false information and make it difficult for traditional media outlets to control the narrative.

The main challenges of news verification include the sheer volume of news content produced and shared online, the difficulty of manually verifying the authenticity of every news item, and the speed at which false information can spread. The sheer volume of news content can make it a resource-intensive task to verify the accuracy of every news item. Moreover, the increasing sophistication of fake news stories means that it can be challenging to detect them without careful analysis. Finally, the speed at which news spreads online means that false information can go viral quickly, making it challenging to stop the spread of fake news once it has begun.

To address these challenges, we propose building a system with online learning that utilizes deep learning techniques such as graph neural networks, convolutional neural networks (CNN), long short-term memory networks (LSTM), gated recurrent units (GRU), and support vector machines (SVM). By automating the detection of fake news using these techniques, the system can adapt to changing trends and patterns, providing a more efficient and effective solution to the problem of news verification. This approach has the potential to improve the accuracy and speed of news verification and reduce the spread of fake news on social media platforms.

This paper presents our approach to fake news detection and provides a detailed description of the system we have built, including its architecture, methodologies, and experimental results. We believe that our approach has the potential to make a significant contribution to the field of news verification and provide a valuable tool for detecting and combating the spread of fake news.

2 Related work

There are not much datasets that Vietnamese researchers can use for fake news detection. One of them are: VFND [1] which consisting of fabricated news articles in the Vietnamese language that were collected between 2017 and 2019 (the year the authors completed their thesis). The articles included in the dataset were classified as genuine or fake based on a variety of sources, including cross-referencing with cited sources or community-based classifications.

Overall speaking, 100% automate fake news detection is still difficult in accuracy and has a long way to go. Fact-checking websites such as Snopes.com, PolitiFact.com, and FactCheck.org depend solely on manual detection by professional experts and organizations. However, this approach is time-consuming, expensive, and requires significant human involvement to maintain the detection systems. Consequently, researchers have developed various automated misinformation detection architectures. One such study by Ozbay and Alatas [2] utilized the term frequency (TF) weighting method and document-term matrix to extract features from texts. They then explored 23 supervised models to identify fake news, and Decision Tree was found to yield the best results. In [3], a novel hybrid fake news detection system was proposed that combines a BERT-based (bidirectional encoder representations from transformers) with a light gradient boosting machine (LightGBM) model. The proposed method was evaluated on three real-world fake news datasets using different word embedding techniques and compared to four different classification approaches. The proposed method outperformed many state-of-the-art methods in detecting fake news based on both headline-only and full-text news content.

One of the most common and effective ways for humans to learn are summaries. Their objective is to produce a representation, which includes the main ideas of the input, while being also shorter than it and which additionally should avoid repetitions. There are a small number of studies on Vietnamese text summarization. Most of these focus on inspecting extractive summarization. The researchers [4] compared a wide range of extractive methods, including unsupervised ranking methods (e.g., LexRank, LSA, KL-divergence), supervised learning methods using TF-IDF and classifiers (e.g., Support Vector Machine, AdaBoost, Learning-2-rank), and deep learning methods (e.g., Convolutional Neural Network, Long-Short Term Memory). This article [5], investigated the combination of pre-trained BERT model and an unsupervised K-means clustering algorithm on extractive text summarization. The authors utilized multilingual and monolingual BERT models to encode sentence-level contextual information and then ranked this information using K-means algorithm.

In fake news classification task, numerous models have been suggested to detect fake news in various types of features and datasets. In the study [6] reported that SVM achieved the highest accuracy of 92% in classification of fake news. The classification models include XG Boost, Random Forest, Naïve Bayesian, KNN, Decision Tree and SVM were used for classification of fake news. In the study of [7], a combined method based on semi-supervised LDA (Linear Discriminant Analysis) and Convolutional Neural Network are used to detect fake news using an unlabeled dataset for the Convolutional Neural Network the unlabeled dataset is labeled. The result of the proposed method of precision is 95.6% and 96.7% recall, which outperforms existing methods for detecting fake news.

3 Dataset

As mentioned earlier, the scarcity of data sources for detecting fake news, particularly in the context of Vietnamese language, poses a significant challenge. Additionally, to support an online learning approach, a continuous influx of updated data is imperative. Therefore, this section delineates the initial phase of our comprehensive data pipeline, focusing on data collection.

The data collection process involves crawling articles from diverse Vietnamese article providers and subsequently categorizing them based on their reliability rates. Three distinct categories have been established:

- Reliable Source Comprising articles from reputable entities such as VTV, Phap luat TPHCM, and Lao dong.

- Unreliable Source: Encompassing articles sourced from platforms like 2sao, aFamily, DaiKyNguyen, and Soha, which may exhibit questionable credibility.

- Mix-type Source: Including articles derived from platforms like Kenh14, ThanhNien, and VNExpress, which display a mixture of reliable and potentially unreliable content.

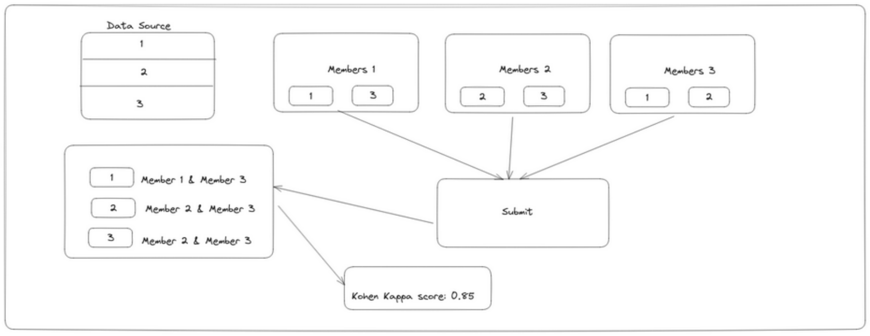

Fig. 1. Guideline for data labeling

Consequently, the collected articles are classified accordingly. Those from reliable sources are labeled as "Verified," while articles originating from unreliable sources are labeled as "Unverified." The mix-type source articles necessitate labeling, and to ensure accuracy, the labeling process is divided into three parts. Each member of the team is tasked with labeling two of these sections.

Upon completion of the labeling process, each part contains labels from two team members. To enhance the reliability of the labeling, a collaborative discussion is undertaken, wherein any mislabeling cases are thoroughly examined and resolved. The Cohen's Kappa formula is applied to calculate the agreement between the two individuals' labels, thereby quantifying the inter-rater reliability.

Through the utilization of the Cohen's Kappa coefficient, an assessment of the consistency and agreement in labeling is achieved. Remarkably, the final calculated value for our labeling process is 0.84, which signifies a notably high level of agreement between the two raters. This outcome attests to the robustness and efficacy of our data collection and labeling approach, laying a solid foundation for subsequent stages of our research in fake news detection and online learning.

The initial stage of the data pipeline involves the crawling of web pages, utilizing the Python libraries 'requests' and 'bs4'. The first step is to extract all categories of articles from the newspaper provider website, such as Chinh tri, Xa hoi, Truyen hinh, and Suc khoe from vtv.vn. Subsequently, each category is explored to extract all available article links. This approach is preferred to simply extracting article links from the homepage of each provider, as the majority of websites have a limited number of articles displayed on their homepage, which typically features only the most popular or trending news

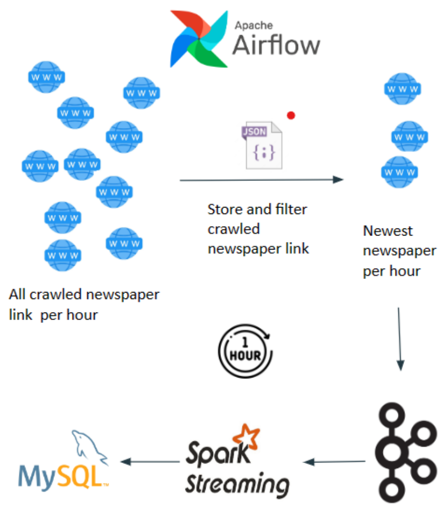
Apache
Airflow
Store and filter crawled newspaper link
All crawled newspaper link per hour
Newest newspaper per hour
1 HOUR
MySQL
Spark Streaming

Fig. 2. Collect data process

During this stage, it is important to note that only a list of article links is obtained. Details such as the author, post date, and content are not yet available. Given that excessive requests to the website server can lead to being blocked, our crawler has been set to restart crawling data every hour. Consequently, there may be numerous duplicate articles and relatively few new ones. To optimize resource utilization and reduce the burden on the Kafka server, we store every extracted newspaper link in a json file and then subsequent crawls come, any links that already exist will be skipped.

Once the list of extracted links has been obtained, a Kafka producer is used to produce the data into a Kafka topic. A Kafka consumer then retrieves the data from the topic and processes it using Spark Streaming. Spark Streaming is utilized to perform text normalization, including the removal of redundant spaces, and to further process the data. The Spark Streaming application processes the incoming data in small batches, allowing for real-time processing of the data stream. By leveraging the power of Spark Streaming, the data can be transformed and analyzed quickly and efficiently, leading to valuable insights and improved data quality

Once the data has been stored in MySQL, it can be easily queried and analyzed using SQL queries or other tools. The ability to store and analyze large volumes of data in real-time enables organizations to make faster and more informed decisions, leading to improved business outcomes and increased competitiveness in today's fast-paced data-driven world.

4 Methods

4.1 Support Vector Machine

Support Vector Machines is a type of supervised learning algorithm which is very powerful in building a classifier. It works by finding a hyperplane in a high-dimensional space that can separate the data into different classes.

Given a labeled training dataset: $(x_1, y_1), ..., (x_n, y_n), x_i \in R^d and y_i \in (-1, 1)$ where x_i is a feature vector representation and y_i is the class label (negative or positive) of a training compound i. We can represent the optimal hyperplane as $w^T * x + b = 0$ where w is the weight vector, x is the input feature vector and b is the bias term. The vector w defines the orientation of the hyperplane, and the scalar b determines its position in the space.

The distance between the hyperplane and the nearest points of each class is known as the margin. The goal of SVMs is to find the hyperplane that maximizes the margin $1/||w^2||$ between the classes. This can be formulated as an optimization problem, where we seek to minimize the norm of the weight vector subject to the constraint that all data points are correctly classified and lie on the correct side of the hyperplane.

To elaborate further, the support vectors are the data points that are closest to the hyperplane and if removed would modify the location of the dividing hyperplane. Thus, support vectors are crucial elements of a data set. In addition, the hyperplane can be thought of as "a line that linearly separates and classifies a set of data" and "the further from the hyperplane our data points lie," the higher the chance that the data points have been accurately classified (Brambrick).

SVMs can also handle non-linearly separable data by mapping the data into a higher-dimensional space using a kernel function. The kernel function allows SVMs to find a hyperplane in the higher-dimensional space that separates the data into different classes. Popular kernel functions include the linear kernel, polynomial kernel, and radial basis function (RBF) kernel.

4.2 Long Short Term Memory

Long Short-Term Memory (LSTM) network are a type of recurrent neural network (RNN) that are designed to deal with the vanishing gradient problem that can occur in traditional RNN. The vanishing gradient problem occurs when the gradients used to update the weights of the network during backpropagation become very small, making it difficult for the network to learn long-term dependencies.

The LSTM network use a series of gates to selectixvely allow information to pass through the network and be stored in a special type of memory called a cell state. The three main gates used in an LSTM network are the input gate, forget gate, and output gate. The input gate determines how much of the new input should be added to the cell state. It takes the previous hidden state h_{t-1} and the current input x_t, and passes them through a sigmoid function to get a

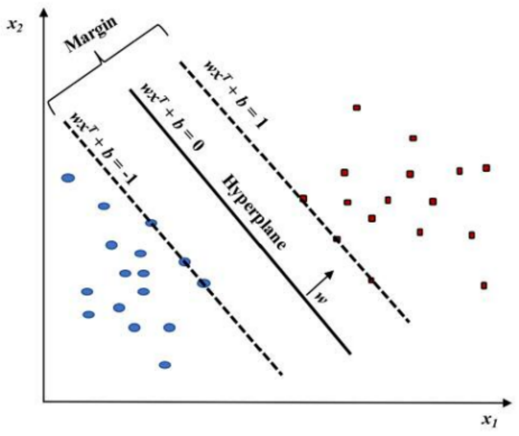

Fig. 3. Linear SVM model

value between 0 and 1:

$$i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i) \tag{1}$$

where i_t is the input gate vector, σ is the sigmoid function, W_i is the weight matrix for the input gate, $[h_{t-1}, x_t]$ is the concatenation of the previous hidden state and current input, and b_i is the bias vector for the input gate. Next, the same information of the hidden state and current state will be passed through the tanh function. It will create a vector $\tilde{C_t} = tanh(W_c * [h_{t-1}, x_t] + b_c)$, which will be used for cell state.

The forget gate determines how much of the previous cell state should be forgotten. It takes the previous hidden state h_{t-1} and the current input x_t, and passes them through a sigmoid function:

$$f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f) \tag{2}$$

where f_t is the forget gate vector, W_f is the weight matrix for the forget gate, and b_f is the bias vector for the forget gate.

The cell state is updated based on the input gate and forget gate vectors. The new cell state C_t is a weighted sum of the previous cell state C_{t-1} and the new input, scaled by the input gate vector i_t and forget gate vector f_t, respectively:

$$C_t = f_t * C_{t-1} + i_t * \tilde{C_t} \tag{3}$$

where W_c is the weight matrix for the cell state update, and b_c is the bias vector for the cell state update.

Finally, the output gate determines how much of the cell state should be output as the new hidden state. It takes the previous hidden state h(t-1) and the current input x(t), and passes them through a sigmoid function:

$$o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o) \tag{4}$$

where o_t is the output gate vector, W_o is the weight matrix for the output gate, and b_o is the bias vector for the output gate. Also, the new hidden state h_t is obtained by applying the output gate to the cell state, scaled by the hyperbolic tangent function: $h_t = o_t * tanh(C_t)$

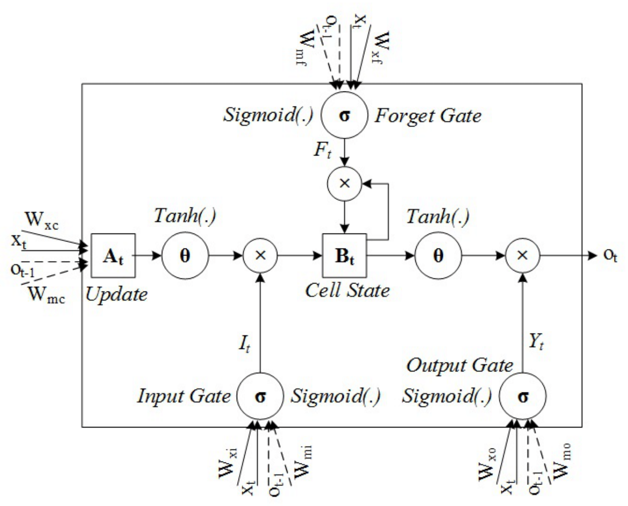

Fig. 4. An LSTM cell containing the input gate, the forget gate, and the output gate

4.3 Gate Recurrent Unit

The Gate Recurrent Unit is a type of recurrent neural network (RNN) that was introduced as a simpler alternative to the Long Short-Term Memory (LSTM) network. Like the LSTMs, the GRUs is designed to deal with the vanishing gradient problem that can occur in traditional RNNs. However, the GRUs achieves this with fewer parameters than the LSTMs, making it a more efficient option.

The GRUs has two gates: an update gate and a reset gate. The special thing about them is that they can be trained to keep information from long ago, without washing it through time or remove information which is irrelevant to the prediction.

The reset gate determines how much of the past information to forget. It takes the previous hidden state h_{t-1} multiplied by its own weight U_z and the current input x_t multiplied by its own weight W_z , and passes them through a sigmoid function to squash the result between 0 and 1:

$$r_t = \sigma(W_r * x_t + U_r * H_{t-1} + b_r) \tag{5}$$

The update gate determines how much of the past information (from previous time steps) needs to be passed along to the future, which is really powerful because the model can decide to copy all the information from the past and eliminate the risk of vanishing gradient problem. It has the same formula as the one for the reset gate:

$$z_t = \sigma(W_z * x_t + U_z * H_{t-1} + b_z) \tag{6}$$

A candidate's hidden state is calculated from the reset gate. This is used to determine the information stored from the past. This is generally called the memory component in a GRU cell. It is calculated by:

$$h'_t = tanh(W * x_t + r_t \odot Uh_{t-1}) \tag{7}$$

where W is weight associated with the current input r_t and U is Weight associated with the hidden layer of the previous timestep h'_t.

Finally, the network needs to calculate h_t — vector which holds information for the current unit and passes it down to the network. That is done as follow:

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h'_t \tag{8}$$

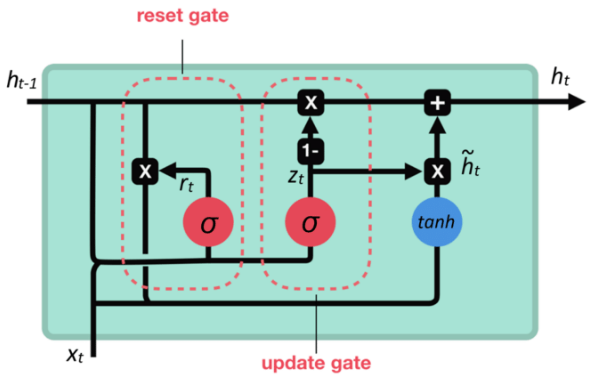

Fig. 5. Gated recurrent unit (GRU) architecture

4.4 Convolutional Neural Network

Convolutional Neural Networks (CNNs) are a type of neural network that are widely used in image and video recognition tasks. The CNN's basic functionality is inspired by the visual cortex of the animal brain, making it a powerful tool for image recognition. However, recent research has shown that CNNs are also effective for text classification. In text classification, the criteria are similar to those for image classification, except that instead of pixel values, we use a matrix of word vectors.

Suppose that we are given a document $D = (w_1, w_2, ...)$ with vocabulary V. In order to feed this data into a Convolutional Neural Network (CNN), we require a vector representation that preserves the internal structure of the data (word order). One common approach is to treat each word as a pixel and represent the document as an image with —D— x 1 pixels and —V— channels. To represent each pixel, we use a one-hot encoding, i.e., a —V—-dimensional vector with all zeros except for a 1 at the index corresponding to the word. For example, if the vocabulary V = “hate”, “He”, “does”, “they”, “look”, and we associate each word with a dimension of the vector in alphabetical order, then the document “He hate they” can be represented as a document vector:

$$x = [00100|10000|00010]^T$$

A CNN typically consists of three layers: a convolutional layer, a pooling layer, and a fully connected layer. The convolutional layer is the core building block of the CNN and is responsible for performing the bulk of the network's computations: extract local features. This layer involves a dot product operation between two matrices: a set of learnable parameters known as a kernel or filter, and a restricted portion of the receptive field.

The pooling layer will capture the most relevant global features with fixed length. The pooling operation is processed on every slice of the representation individually. There are several pooling functions but the most popular process is max pooling, which reports the maximum output from the neighborhood.

The last layer is fully connected, where a softmax classifier is applied to predict the probability distribution over categories, since we are focusing on text classification. The network is trained with the objective that minimizes the cross-entropy of the predicted distributions and the actual distributions.

4.5 Graph Neural Network

Graph Neural Networks are a type of neural network that can be used to model and analyze data that is represented as graphs. As implied by the term – Graph Neural Networks – the most essential component of GNNs is a graph. Graphs can be used to represent a wide range of real-world phenomena, such as social networks, protein structures, and transportation systems. We can consider a graph that depicts a city network: cities are depicted as nodes, while the highways that link them are depicted as edges. We may use the graph network to solve a variety of issues relating to these cities, such as determining which cities are well-connected or determining the shortest distance between two cities.

Mathematically, let represent a graph as $G = (V, E)$ where V is the set of nodes or vertices, and E is the set of edges or links between the nodes. Each node $v \in V$ can be associated with a feature vector $X_v \in R^d$, which encodes information about the node. Similarly, each edge $e \in E$ can be associated with a feature vector $X_e \in R^d$, which encodes information about the relationship between the nodes it connects. There are two tasks of interest: Node classification, where each node $v \in V$ has an associated label y_v and the goal is to learn a

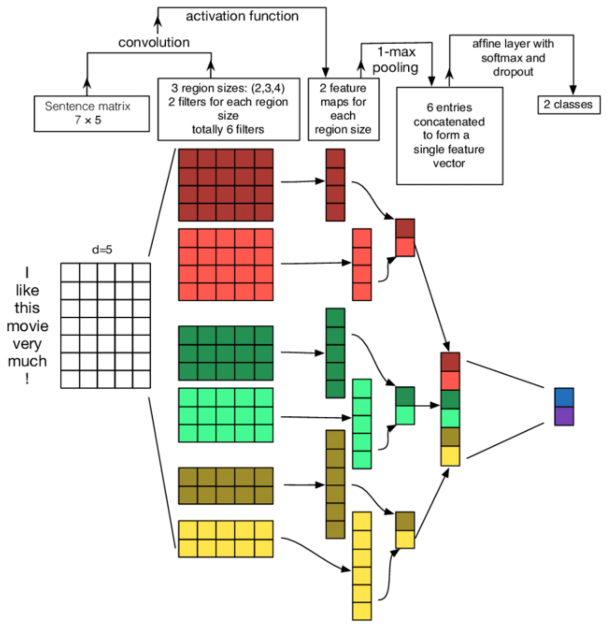
activation function
convolution
1-max pooling
affine layer with softmax and dropout
Sentence matrix 7 × 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 entries concatenated to form a single feature vector
2 classes
d=5
I
like
this
movie
very
much
!

Fig. 6. CNN architecture for (short) document classification

representation vector h_v of v such that v's label can be predicted as $y_v = f(h_v)$; Graph classification, where, given a set of graphs $\{G_1, ..., G_N\} \subseteq G$ and their labels $\{y_1, ..., y_N\} \subseteq y$, we aim to learn a representation vector h_G that helps predict the label of an entire graph, $y_G = g(h_G)$.

GNNs use the graph structure and node features X_v to learn a representation vector of a node, h_v, or the entire graph, h_g. Mathematically, we have: $f(G, X_v, X_e) = h_v$. Modern GNNs follow a neighborhood aggregation strategy, which update a node's representation by aggregating its neighbors' representations iteratively. After k iterations, the node's representation captures the structural information within its k-hop network neighborhood. Formally, the k-th layer of a GNN is:

$$a_v{}^{(k)} = AGGREGATE^{(k)}(\{h_u{}^{(k-1)} : u \in N(v)\})$$
$$h_v{}^{(k)} = COMBINE^{(k)}(h_v{}^{(k-1)}, a_v{}^{(k-1)})$$

where $h_v{}^{(k)}$ is the feature vector of node v at the k-th iteration/layer, and $N(v)$ is a set of nodes adjacent to v. The choice of $AGGREGATE^{(k)}(\cdot)$ and $COMBINE^{(k)}$ in GNNs is crucial too. Diverse architectural proposals have been suggested to address the issue of AGGREGATE. Graph Convolutional Networks have adopted the use of element-wise mean pooling instead of AGGREGATE, with the AGGREGATE and COMBINE procedures being seamlessly integrated in the following manner.:

$$a_v{}^{(k)} = RELU(W * MEAN\{h_u{}^{(k-1)} : \forall u \in N(v) \cup \{v\}\})$$

where W is a learnable matrix.

In the context of node classification, the node representation $h_v{}^K$ of final iteration is leveraged for prediction. Conversely, for graph classification, the READOUT function serves to amalgamate node characteristics from the concluding iteration, culminating in the derivation of the comprehensive graph representation h_G:

$$h_G = READOUT(\{h_v{}^K | v \in G\})$$

with READOUT can be a simple permutation invariant function such as summation or a more sophisticated graph-level pooling function.

5 Experiments

5.1 System Design

The present research endeavors to design and implement an end-to-end system for the verification of Vietnamese news articles. The proposed system is engineered to enhance user experience by reducing input effort through the use of a user-friendly extension. Upon pasting a news link into the extension, the system generates predictions, and users have the option to modify the results before submitting them. The news articles are processed through a series of stages within the system.

Initially, the system incorporates the newspaper3k framework to crawl and extract the content and title of the news articles. The extracted content is then subjected to the vit5-large-vietnews-summarization service, provided by VietAI, to generate concise summaries, which are subsequently sent back to the News Collection system. Moreover, the summarized content undergoes news classification and named entity recognition using the underthesea library. The identified news category and mentioned entities are subsequently integrated into the News Collection System for further processing.

To manage the incoming news articles, the system employs Kafka as an intermediary, facilitating the transfer of data to the database. Upon accumulating a substantial number of news samples (exceeding 1000), the system signals the Kafka server to initiate the ML Training service. This service trains models on the collected data, and the model exhibiting the highest evaluation score is selected for subsequent predictions. The system boasts continuous learning capabilities, ensuring its adaptability to the ever-changing landscape of global news.

News articles arriving from both user extensions and the web crawler follow a similar flow, traversing through Kafka before being processed by the News Collection module. The news undergo summarization and classification services before being assembled into the Fit data table, which prepares the news for model training. The system conducts an iterative process, looping through each news article from the News table in the database, identifying articles sharing the same category and mentioned entities. By applying SparkSQL of Spark, the News Revelancy service gauges the relevance of the news articles, leveraging the Sbert model for enhanced accuracy. Through SparkSQL, the system counts the number of news articles with revelancy results exceeding 0.6 and possessing verified labels, culminating in the generation of the Fit Data table within the MySQL database. Preprocessing and fitting processes are carried out using PySpark to train and optimize four models. To ensure seamless integration and deployment, all services are containerized through Docker, consolidated within the UIT-News-Verification docker-compose network.

In conclusion, the comprehensive end-to-end system for news verification demonstrates a scientific and practical approach to address the challenges posed by misinformation in the Vietnamese news landscape. The incorporation of state-of-the-art technologies, continuous learning, and efficient data processing allows the system to accurately verify news articles, facilitating trustworthy and reliable information dissemination.

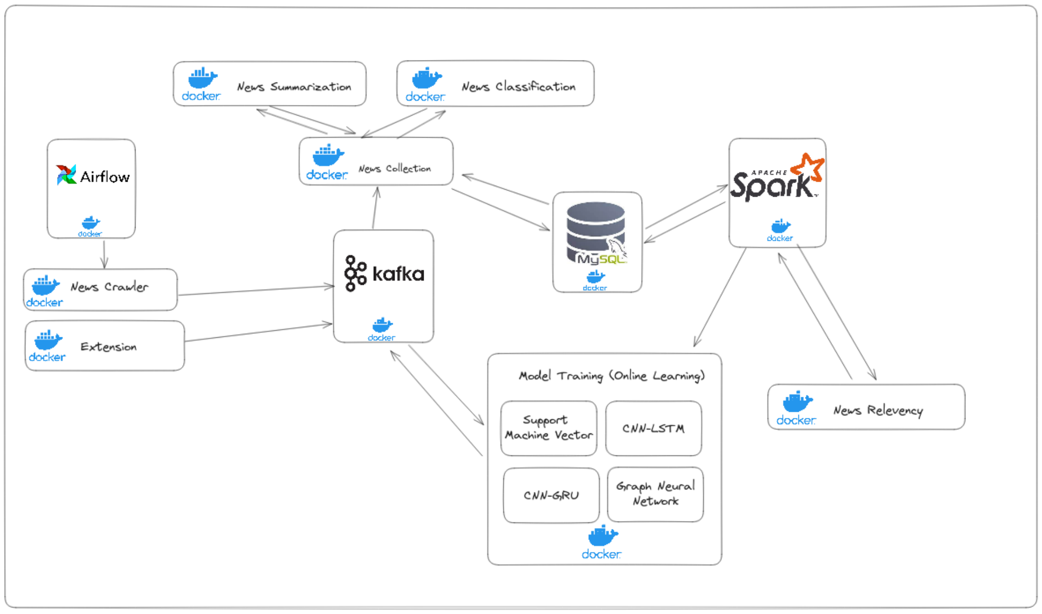

Fig. 7. System Design for News Verification system

5.2 Feature Engineering

Feature engineering is a critical aspect of our comprehensive end-to-end system for news verification, enabling the evaluation and verification of news articles through various distinct dimensions. Leveraging cutting-edge techniques, we have devised three essential categories of feature engineering: Style-based article verification, Source-based article verification, and Media Diverse article verification.

Style-based Article Verification Style-based article verification entails the extraction and analysis of specific linguistic and writing patterns that are characteristic of genuine news sources. By capturing features such as sentence structure, tone, vocabulary, and grammar, our system can discern the stylistic attributes unique to credible news articles. Utilizing natural language processing (NLP) techniques, including syntactic parsing and sentiment analysis, we can identify irregularities or inconsistencies in writing style that may indicate potential misinformation. Moreover, the system employs statistical measures to assess the coherence and readability of news content, aiding in the differentiation between genuine and fabricated articles.

Source-based Article Verification The Source-based article verification component of our system centers on scrutinizing the credibility and reputation of the news sources themselves. To achieve this, the system relies on a database of verified and reputable news outlets. By cross-referencing the origin of incoming news articles with this trusted source repository, our system can ascertain the reliability of the article's source. This verification process is further augmented by employing algorithms that evaluate the historical accuracy and editorial practices of news outlets, thus providing users with valuable insights into the credibility of the news source.

Media Diverse Article Verification The Media Diverse article verification aspect is designed to mitigate the impact of information echo chambers by promoting media diversity. Our system is equipped with a sophisticated algorithm that assesses the diversity of media sources referenced within a news article. By comparing the article's sources against an extensive and representative media database, we can evaluate whether the news article draws from a broad spectrum of perspectives and viewpoints. The presence of diverse sources enhances the credibility of the article and contributes to a comprehensive understanding of the topic at hand. Consequently, our system empowers users to make informed decisions based on a balanced range of information.

Collectively, the three pillars of feature engineering within our news verification system provide a multi-dimensional assessment of news articles, ensuring a robust and thorough verification process. The integration of NLP techniques, source validation, and media diversity evaluation fosters a comprehensive understanding of news credibility. By empowering users to distinguish between credible and unreliable information, our feature engineering approach strengthens the overall integrity and reliability of news dissemination in the Vietnamese landscape. As part of our ongoing commitment to continuous learning, we continually refine and enhance these feature engineering methods to adapt to emerging challenges in the ever-evolving world of digital information.

5.3 Evaluation metrics

Precision, recall, and F1 score are commonly used metrics in machine learning and information retrieval to evaluate the performance of classification or retrieval algorithms. These metrics are particularly useful in binary classification problems where the goal is to predict whether a given instance belongs to a positive class or a negative class. To understand these metrics, we first need to define some terms. Let's assume we have a binary classification problem where the goal is to predict whether a given instance belongs to a positive class or a negative class. In this case, we can define the following:

- True positives (TP): the number of instances that are correctly predicted as positive.
- False positives (FP): the number of instances that are incorrectly predicted as positive when they actually belong to the negative class.
- True negatives (TN): the number of instances that are correctly predicted as negative.
- False negatives (FN): the number of instances that are incorrectly predicted as negative when they actually belong to the positive class.

Precision Precision is a fundamental metric used to evaluate the performance of a classification model, particularly in binary classification tasks. It quantifies the proportion of true positives among all instances that are predicted as positive. In other words, it measures how precise the classifier is when it predicts that an instance belongs to the positive class. A high precision indicates that the classifier is very good at identifying positive instances, while a low precision indicates that the classifier is making many false positive predictions. Mathematically, it is defined as:

$$precision = \frac{TP}{TP + FP} \tag{9}$$

Recall Recall measures the proportion of true positives among all instances that actually belong to the positive class. In other words, it measures how well the classifier is able to identify all positive instances. A high recall indicates that the classifier is very good at identifying all positive instances, while a low recall indicates that the classifier is missing many positive instances. It can be defined in math formula as below:

$$recall = \frac{TP}{TP + FN} \tag{10}$$

F1 score F1 score is a harmonic mean of precision and recall. It is a balanced measure that takes both precision and recall into account. F1 score is particularly useful when we want to compare classifiers that have different precision and recall values. A high F1 score indicates that the classifier is good at both precision and recall. Mathematically, its formula is:

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{11}$$

It is important to note that precision and recall are complementary metrics. In general, increasing precision leads to a decrease in recall, and vice versa. This is because increasing the threshold for positive predictions (i.e., making the classifier more selective) increases precision but decreases recall, while decreasing the threshold (i.e., making the classifier less selective) increases recall but decreases precision. It is important to not only define these metrics but also to discuss their relevance and limitations. For example, precision may be more important in some applications (e.g., medical diagnosis) where false positives can have serious consequences, while recall may be more important in other applications (e.g., spam filtering) where false negatives are more tolerable.

5.4 Experimental Results

After obtaining the initial dataset, we proceeded to train it using four different models. The results of the training process are presented in Table 1, showcasing the performance metrics of precision, recall, and F1 score for each model.

Table 1. The results of evaluating at the initial state.

Methods	Precision	Recall	F1 score
SVM	0.55	0.8	0.65
GNN	0.7	0.85	0.77
CNN	0.65	0.8	0.72
CNN-GRU	0.72	0.78	0.75
CNN-LSTM	0.73	0.74	0.73

From the table, it can be observed that the SVM (Support Vector Machine) model achieved a precision of 0.55, a recall of 0.8, and an F1 score of 0.65. The GNN (Graph Neural Network) model demonstrated improved performance with a precision of 0.7, a recall of 0.85, and an F1 score of 0.77. The CNN (Convolutional Neural Network) model exhibited a precision of 0.65, a recall of 0.8, and an F1 score of 0.72.

Additionally, the CNN-GRU (Convolutional Neural Network - Gated Recurrent Unit) model delivered notable results with a precision of 0.72, a recall of 0.78, and an F1 score of 0.75. Lastly, the CNN-LSTM (Convolutional Neural Network - Long Short-Term Memory) model showcased commendable performance with a precision of 0.73, a recall of 0.74, and an F1 score of 0.73.

Based on the table, it is evident that each model exhibits distinct strengths and weaknesses in terms of precision, recall, and F1 score. The GNN model achieved the highest recall and F1 score, indicating its effectiveness in identifying true positive instances, while the CNN-LSTM model demonstrated the highest precision, implying its proficiency in minimizing false positive predictions.

We test the whole pipeline with in 2 turn of model training we got the following results

Table 2. The results of evaluating at the first turn of model training.

Methods	Precision	Recall	F1 score
SVM	0.53	0.6	0.56
GNN	0.81	0.83	0.82
CNN	0.66	0.83	0.73
CNN-GRU	0.70	0.71	0.70
CNN-LSTM	0.73	0.72	0.72

Table 3. The results of evaluating at the second turn of model training.

Methods	Precision	Recall	F1 score
SVM	0.52	0.61	0.56
GNN	0.8	0.82	0.81
CNN	0.64	0.81	0.71
CNN-GRU	0.72	0.71	0.72
CNN-LSTM	0.73	0.712	0.72

Comparing the three tables, we can see that the models' performances remained relatively consistent between the first and second turns of training. While some minor fluctuations in F1 scores were observed, the overall ranking of the models in terms of performance remained unchanged, with GNN leading the pack followed by CNN, CNN-GRU, CNN-LSTM, and SVM.

In conclusion, the evaluation of the models in two turns of model training provides valuable insights into their performance stability and effectiveness in the task of fake news detection. Further optimizations and fine-tuning can be undertaken based on these results to enhance the overall performance of the pipeline.

6 Conclusion

This research presents the development of a comprehensive end-to-end system for news verification. The efficacy of this system is rigorously tested through two rounds of model training. The experimental results demonstrate the stability and adaptability of the models to evolving contexts. Nonetheless, the system's performance may be susceptible to potential inaccuracies if users collectively provide incorrect labels. To address this limitation, we propose future enhancements wherein the system will incorporate a ranking mechanism to prioritize user answers based on their historical accuracy in labeling news through the provided extension. This refinement aims to further improve the overall reliability and precision of the news verification process.

References

1. H. Q. Thanh and ninh-pm se, "thanhhocse96/vfnd-vietnamese-fake-news-datasets: Tp hp các bài báo ting Vit và các bài post Facebook phân loi 2 nhãn Tht & Gi (228 bài)," Feb. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2578917
2. F. A. Ozbay and B. Alatas, "Fake news detection within online social media using supervised artificial intelligence algorithms," *Physica A: statistical mechanics and its applications*, vol. 540, p. 123174, 2020.
3. E. Essa, K. Omar, and A. Alqahtani, "Fake news detection based on a hybrid bert and lightgbm models," *Complex & Intelligent Systems*, pp. 1–12, 2023.
4. M.-T. Nguyen, H.-D. Nguyen, T.-H.-N. Nguyen, and V.-H. Nguyen, "Towards state-of-the-art baselines for vietnamese multi-document summarization," in *2018 10th International Conference on Knowledge and Systems Engineering (KSE)*, 2018, pp. 85–90.
5. H. Q. To, K. Van Nguyen, N. L.-T. Nguyen, and A. G.-T. Nguyen, "Monolingual versus multilingual bertology for vietnamese extractive multi-document summarization," *arXiv preprint arXiv:2108.13741*, 2021.
6. I. Vogel and M. Meghana, "Detecting fake news spreaders on twitter from a multilingual perspective," in *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 2020, pp. 599–606.
7. R. Mansouri, M. Naderan-Tahan, and M. J. Rashti, "A semi-supervised learning method for fake news detection in social media," in *2020 28th Iranian Conference on Electrical Engineering (ICEE)*. IEEE, 2020, pp. 1–5.
8. E. Tacchini, G. Ballarin, M. L. Della Vedova, S. Moret, and L. De Alfaro, "Some like it hoax: Automated fake news detection in social networks," *arXiv preprint arXiv:1704.07506*, 2017.
9. A. Kansal, "Fake news detection using pos tagging and machine learning," *Journal of Applied Security Research*, vol. 18, no. 2, pp. 164–179, 2023.
10. H. Rashkin, E. Choi, J. Y. Jang, S. Volkova, and Y. Choi, "Truth of varying shades: Analyzing language in fake news and political fact-checking," in *Proceedings of the 2017 conference on empirical methods in natural language processing*, 2017, pp. 2931–2937.
11. K. Wang, Y. Ding, and S. C. Han, "Graph neural networks for text classification: A survey," *arXiv preprint arXiv:2304.11534*, 2023.
12. L. Huang, D. Ma, S. Li, X. Zhang, and H. Wang, "Text level graph neural network for text classification," *arXiv preprint arXiv:1910.02356*, 2019.
13. N. Andhale and L. A. Bewoor, "An overview of text summarization techniques," in *2016 international conference on computing communication control and automation (ICCUBEA)*. IEEE, 2016, pp. 1–7.
14. M.-T. Nguyen, H. Diep, T.-H.-N. Nguyen, and V.-H. Nguyen, "Towards state-of-the-art baselines for vietnamese multi-document summarization," 11 2018.