Evaluation of the Keyword Selection Methods Effectiveness for the Fake News Classification

Khrystyna Lipianina-Honcharenko[1,†], Dmytro Lendiuk[1,†], Nazar Melnyk[1,†], Myroslav Komar[1,†], Taras Lendiuk[1,*,†]

[1] *West Ukrainian National University, Lvivska str., 11, Ternopil, 46000, Ukraine*

Abstract

This study is devoted to the comparison of different methods of choosing keywords for the classification of fake and true news based on Ukrainian and Russian datasets. TF-IDF, RAKE, Yake!, LSA, LDA and TextRank methods were used, which were evaluated by such metrics as accuracy, recall, precision and F1-measure. The results showed that the TF-IDF and RAKE methods are the most effective for news classification, showing an overall accuracy of 88% and 87.94%, respectively. The TF-IDF method achieved precision for fake news at the level of 0.90 and recall for true news at the level of 0.94. Other methods such as Yake! and LDA, showed significantly lower accuracy — 78% and 67%, respectively. The obtained results indicate the prospects of using the TF-IDF and RAKE methods for the development of a system for fake news automatic detection.

Keywords

news classification, fake news, TF-IDF, RAKE, machine learning, accuracy, disinformation detection

1. Introduction

The problem of disinformation, in particular the spread of fake news, is gaining more and more importance in the modern information environment. This problem is especially acute in the conditions of hybrid conflicts and information warfare, where false information is used as a tool to influence public opinion and social stability. Therefore, the development of effective methods for the fake news automatic detection is an extremely important task for ensuring objective information to society. In this context, the use of machine learning and text-based methods for processing large volumes of data opens up new opportunities for automating the process of false information identifying.

This study focuses on the comparison of different keyword selection methods for news classification, such as TF-IDF, RAKE, Yake!, LSA, LDA, and TextRank. The main goal is to identify the most effective approaches that can be used to create a fake news detection tool. With the help of experiments using sets of news data, the effectiveness of each method was evaluated according to such indicators as accuracy, recall, precision and F1-measure. The results of this study will contribute to the further development of automatic disinformation detection technologies based on machine learning.

This work focuses on the comparison of keyword selection methods for the classification of fake and true news and consists of several chapters. Chapter 2 analyzes existing research in the field of disinformation detection and text processing methods. Chapter 3 describes the research

Information Technology and Implementation (IT&I-2024), November 20-21, 2024, Kyiv, Ukraine

* Corresponding author.

† These authors contributed equally.

xrustya.com@gmail.com (Kh. Lipianina-Honcharenko); dmytrolenduk@gmail.com (D. Lendiuk); 88nazar88@gmail.com (N. Melnyk); mko@wunu.edu.ua (M. Komar); tl@wunu.edu.ua (T. Lendiuk)

0000-0002-2441-6292 (Kh. Lipianina-Honcharenko); 0009-0004-2728-7889 (D. Lendiuk); 0009-0000-5917-1099 (N. Melnyk); 0000-0001-6541-0359 (M. Komar); 0000-0001-9484-8333 (T. Lendiuk)

methodology, including data collection processes, pre-processing and application of algorithms for keyword extraction and construction of classification models. Chapter 4 provides an analysis of the obtained results with an assessment of the accuracy of the methods using the precision, recall and F1-measure metrics. The final section 5 highlights the effectiveness of the TF-IDF and RAKE methods and provides recommendations for further development of the system for automatic detection of fake news.

2. Related Work

Several studies have investigated methods for detecting fake news by comparing machine learning (ML) and deep learning (DL) approaches. For example, the study of Khalil Ur et al. [1] compared SVM (ML) and LSTM (DL), where LSTM achieved an accuracy of 99.54%, while SVM showed 99.29% in authenticating news articles. Apparently, Chauhan and Palivela [2] noted the high performance of neural networks, particularly LSTM, which achieved 99.88% accuracy.

These studies often use approaches such as Random Forest, Decision Tree, and feature extraction methods such as TF-IDF. For example, in a study by Johnson et al. [3] Random Forest achieved 100% accuracy, outperforming the Decision Tree method which achieved 93.64%.

The study [4] examines three feature extraction techniques for detecting fake news: TF-IDF, Count Vectorizer (CV), and Hash Vectorizer (HV). These methods use linguistic features to improve the disinformation identification.

The study [5] proposes a new approach to detecting fake news based on the use of Positive and Unlabeled Learning (PUL) with the addition of an attention mechanism to identify the most relevant keywords in a news network. The GNEE algorithm based on Graph Attention Networks (GAT) is used, which allows automatic selection of important terms for news classification. Results showed a 2–10% improvement in F1-measure in scenarios with limited labeled data (only 10% of fake news was labeled).

Most comparisons show the benefits of deep learning, especially using LSTM networks, due to their ability to better handle complex text patterns, while traditional ML models like SVM continue to provide robust results. The main difference of this study is a thorough comparative analysis of keyword extraction methods, such as TF-IDF, RAKE, Yake!, KeyBERT, LSA, LDA and TextRank, combined with classifiers, aimed at a specific geopolitical context using Ukrainian and Russian datasets, which increases its significance for detecting fake news during the Russian-Ukrainian war.

Considering this, the purpose of this study is to compare and determine the most effective methods of choosing keywords for the classification of fake and true news, which will allow creating an accurate tool for disinformation automatic detection. This study also aims to analyze different text processing approaches, such as TF-IDF, RAKE, Yake!, LSA, LDA, and TextRank, in order to evaluate their impact on classification accuracy and identify the most effective ones for further implementation in the fake news detection system.

3. Research Methodology

3.1. Research architecture

Figure 1 illustrates the architecture of the study, which is aimed at a comparative analysis of text processing methods for detecting fake news. At the first stage, data is collected and prepared, where each text document goes through the processes of tokenization, removal of stop words, and bringing the text to the lower case. After preprocessing, the data is fed to a keyword extraction unit that applies various methods to identify the most relevant terms in the text, including TF-IDF, RAKE, Yake!, KeyBERT, LSA, LDA, and TextRank. Each of these methods offers a unique approach to evaluating the importance of words and phrases in a document.

The next stage involves using the received keywords to build classification models. The purpose of this step is to evaluate how effectively each keyword extraction method can help detect fake news. At the final stage, the model is evaluated, where accuracy, completeness, F1-measure and other metrics are measured for each method. This approach allows for a comprehensive analysis of various text processing techniques and their impact on the accuracy of fake news classification.

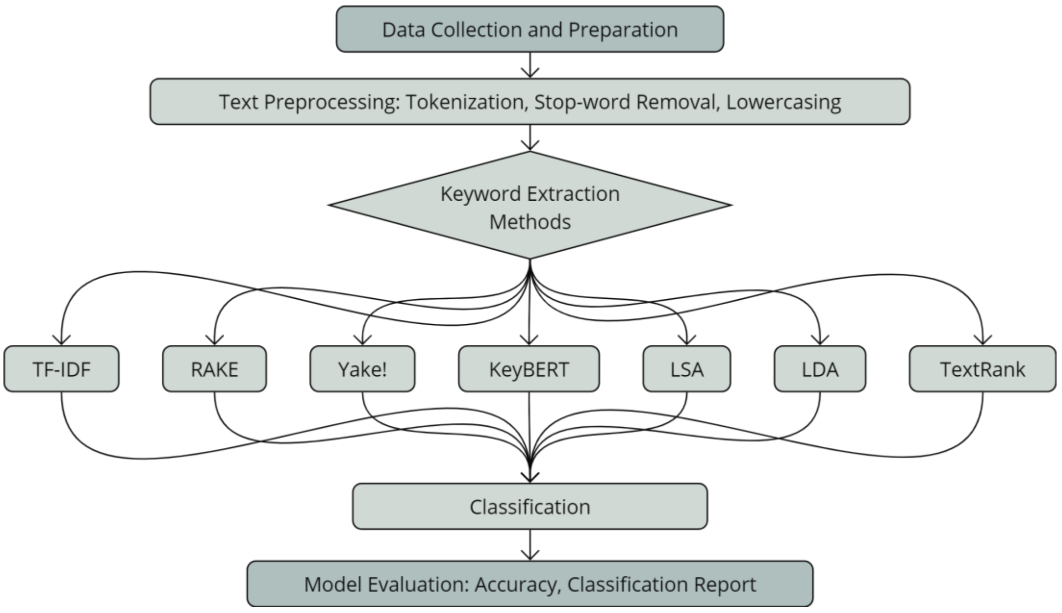

Data Collection and Preparation
Text Preprocessing: Tokenization, Stop-word Removal, Lowercasing
Keyword Extraction Methods
TF-IDF
RAKE
Yake!
KeyBERT
LSA
LDA
TextRank
Classification
Model Evaluation: Accuracy, Classification Report

Figure 1: Structure of the research architecture

3.2. Datasets description

Ukrainian News Fake vs True Dataset and Fake News UA Dataset contain news from Ukrainian and Russian sources, collected for the purpose of classifying fake and authentic news. These datasets are important for information space analysis and machine learning research related to detecting fake news during the Russian-Ukrainian war [6].

The Ukrainian News Fake vs True Dataset contains approximately 10,700 news headlines collected from Ukrainian and Russian Telegram channels between February 24 and December 11, 2022, during a full-scale Russian-Ukrainian invasion [7]. This open data set includes both authentic news and fake news published by Russian sources. In particular, 4,522 titles are classified as fake, and 56,237 as authentic. Two types of labels are used to classify news: "True" for verified news and "False" for fake news. Data sources include Telegram channels such as Suspilne Novyny, Perepichka NEWS, and NR, as well as disinformation channels, including Vox Ukraine and War on Fakes. The dataset was developed for a university project to classify news and is useful for machine learning research.

Fake News UA Dataset contains news samples collected from Ukrainian and Russian sources for the purpose of classifying fake news. The dataset contains various articles and news headlines from Ukrainian Telegram channels, as well as links to primary sources such as Vox Ukraine, Ukrainian Pravda, Espreso, and Radio Svoboda. Each record contains the text of the news, its original source, language (Ukrainian or Russian), as well as a label indicating whether the news is fake. In total, there are about 12,749 news in the set, which have the labels "Fake" (3,375 news) or "True". The data set is intended for text classification tasks and also contains time stamps that allow to investigate the dynamics of the spread of fake news.

Analysis of the graph (Figure 2) of the distribution of "Fake" and "True" classes after removing records with omissions shows an uneven distribution of news between fake and true in the combined data set. In general, the number of true news (True) is 4,668 (about 58% of the total), which exceeds the number of fake news (Fake), which is only 3,375 (about 42%).

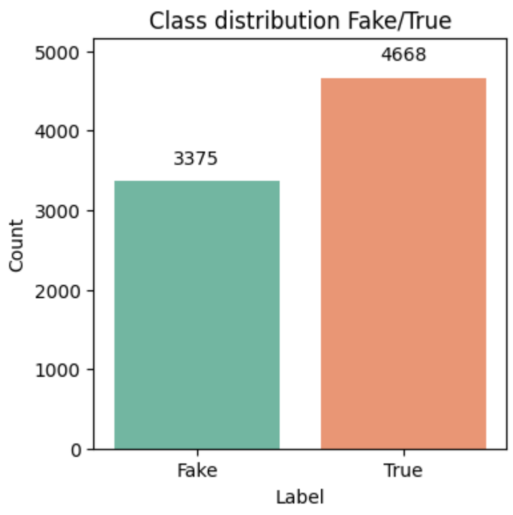


Figure 2: Class distribution

3.3. Description of the used keywords selection

3.3.1. TF-IDF

TF-IDF (Term Frequency-Inverse Document Frequency) — is a method for evaluating the importance of a term in a document relative to an entire document collection or corpus. It consists of two main parts: TF (Term Frequency) and IDF (Inverse Document Frequency), which together help reveal how important is a particular term in a particular document compared to the entire set of documents [8].

Term frequency shows how often a certain term occurs in a particular document. The formula for calculating TF looks like this:

$$TF(t,d) = \frac{f(t,d)}{\sum_k f(k,d)},$$

where: $f(t,d)$ — the number of occurrences of the term ttt in the document d, $\sum_k f(k,d)$ — the total number of all terms in the document d.

Thus, TF shows the share of a certain term from the total number of terms in the document.

The inverse frequency of a document indicates the importance of a term in the entire corpus of documents. If a term occurs in many documents, its importance decreases. The formula for IDF is as follows:

$$IDF(t,D) = \frac{\log(|D|)}{(|d \in D: t \in d|)},$$

where, $|\ D\ |$ — the total number of documents in the corpus, $|\ \{d \in D: t \in d\}\ |$ — the number of documents that contain the term t.

The less frequently a term appears in documents, the higher its IDF value will be.

The final TF-IDF formula combines both of these metrics:

$$TF - IDF(t,d,D) = TF(t,d) \cdot IDF(t,D).$$

This means that the importance of a term depends both on its frequency in the document and on its overall prevalence in the corpus of documents. A term that occurs frequently in one document but rarely in others will have a high TF-IDF value. In contrast, terms that occur in many documents will have a low TF-IDF value, even if they occur frequently in a particular document.

3.3.2. RAKE

RAKE (Rapid Automatic Keyword Extraction) — is a method for automatically extracting keywords from text based on the frequency of terms and their relationship with other terms in the document [9]. It does not require prior training on data or linguistic resources (such as dictionaries or lexical bases), making it efficient and fast for identifying relevant keywords.

The text is divided into phrases consisting of one or more words. For this, punctuation marks (.,!?), stop words (general words that do not carry a specific semantic load) and other symbols that are not part of phrases that can be keywords are used. These phrases can consist of one or more words standing next to each other.

For each term in the candidate phrases, two main indicators are calculated:

- Term frequency f(t): number of occurrences of the term in the entire text.
- Term degree d(t): the number of all terms with which this term is in the same phrase. This can be thought of as the sum of the number of words in all phrases containing that term.

The importance of a term is calculated based on the ratio of its degree to frequency:

$$Score(t) = \frac{f(t)}{d(t)}.$$

This shows the relationship between the number of terms that appear together with a given term and the number of times it appears in the text. The higher the degree of the term relative to its frequency, the more important this word is for the formation of the key phrase.

For each candidate phrase, its total score is calculated as the sum of scores of all terms included in the phrase:

$$Score(Phrase) = \sum_{t \in Phrase} Score(t)\ .$$

Thus, phrases containing important terms receive high scores and are considered as potential key phrases for the text.

3.3.3. Yake!

YAKE! (Yet Another Keyword Extractor) — is a method for automatically extracting keywords from text, which is based on the statistical characteristics of terms in the text without the need for training on external corpora or using linguistic resources [10]. The main idea of YAKE! is that it uses several metrics to evaluate the importance of each term in the text, taking into account the local context, position, and prevalence of the term.

First, the text is broken down into candidate terms and phrases using punctuation marks, stop words, and other symbols that do not belong to key phrases.

The frequency of the term in the text is calculated, that is, the number of times the term appears in the text. The more often a term appears in a document, the higher its frequency score:

$$TF(t) = f(t, d),$$

where $f(t, d)$ — the number of occurrences of term t in document d.

YAKE! takes into account the position at which the term appears in the text. A term that appears closer to the beginning of a text may have a different weight than one that appears closer to the end.

YAKE! takes into account whether the term appears in different contexts (phrases). If a term occurs in only a limited number of contexts, its importance diminishes.

The relationship of the term with other terms in the candidate phrases is evaluated. If a term appears frequently with the same other terms, its uniqueness is reduced.

The final formula for evaluating a key phrase includes a weighted consideration of term frequency, position, prevalence, and uniqueness:

$$Score(t) = W1 \cdot f(t, d) + W2 \cdot Pos(t, d) + W3 \cdot Spread(t),$$

where: $f(t,d)$ — frequency of term ttt, $Pos(t,d)$ — the position of the term ttt in the document, $Spread(t)$ — the number of contexts in which the term appears, $W1, W2, W3$ — weighting factors to adjust the influence of each component.

3.3.4. KeyBERT

KeyBERT — is a method for extracting keywords from text based on the use of transformer models [11], in particular the BERT (Bidirectional Encoder Representations from Transformers) model. Unlike statistical methods such as TF-IDF or RAKE, KeyBERT uses semantic understanding of text to identify key phrases and words, allowing it to take into account the context of terms in the text.

First, the text is fed as input to the BERT model, which generates multidimensional vector representations (embeddings) for each term in the text. These vectors reflect the semantic properties of words and phrases, which allows us to evaluate their similarity.

KeyBERT uses a vector representation of the entire document or text to extract keywords. The vectors for each word or phrase in the text are then compared to the document vector to determine the most relevant terms. If the phrase or word vector is closest to the document vector, that phrase or word is considered a potential keyword.

Mathematically, the similarity between a word vector w and a document vector d is calculated using cosine similarity:

$$cosine_{similarity(w,d)} = \frac{w \cdot d}{\|w\|\|d\|}.$$

The higher the cosine similarity, the more the word or phrase matches the content of the document.

KeyBERT calculates the cosine similarity between the vectors of all phrases/terms and the document vector. Those phrases with the highest similarity are chosen as keywords. This allows not only to evaluate terms by their frequency or statistical indicators, but also to take into account the semantic context of the text.

KeyBERT supports the extraction of multi-word phrases (n-grams), which allows it to detect not only individual words, but also important phrases that may carry more meaning.

3.3.5. LSA

LSA (Latent Semantic Analysis) — is a text processing method used to identify hidden semantic relationships between terms in text documents [12, 13]. The main goal of LSA is to reduce the dimensionality of the space of documents and words to reveal relationships between terms that are not obvious at the surface level.

The text corpus is first transformed into a matrix of term-documents. Each row of the matrix corresponds to a specific term, and each column corresponds to a document. The values in the cells can be the number of occurrences of the terms in the documents or weighted values such as TF-IDF. Thus, the initial representation of the text is high-dimensional and sparse.

Formally, let A be an m×n term-document matrix, where mmm is the number of terms and n is the number of documents.

LSA applies singular value decomposition (SVD) to reduce the dimensionality of the original matrix. SVD decomposes the AAA matrix into three matrices:

$$A = U\Sigma V^T,$$

where: U — orthogonal matrix of terms, Σ — diagonal matrix of singular values containing ranked singular values (from largest to smallest), V^T — orthogonal matrix of documents.

The singular values in the matrix Σ determine the importance of each of the vectors in U and V^T. Smaller singular values can be discarded, reducing the dimensionality of the space, leaving only the most significant components.

After performing SVD, only the largest singular values and their corresponding components in the U and V^T matrices are selected. This allows reducing the dimensionality of the matrix, leaving only the main latent semantic structures of the text.

Thus, the reduced dimensionality matrix makes it possible to analyze documents and terms in a new latent space, where terms with similar contexts will be close to each other.

In the reduced latent variable space, terms that often occur in similar contexts but may not be obvious on the surface become closer together. This helps discover hidden semantic relationships between terms and documents, which allows LSA to work effectively in the tasks of topic modeling, information retrieval, and text classification.

3.3.6. LDA

LDA (Latent Dirichlet Allocation) — is a statistical topic modeling method used to automatically extract hidden topics from a large collection of text documents [14]. The basic idea of LDA is that each document is treated as a mixture of several topics, and each topic is treated as a distribution of terms. The goal of the algorithm is to identify these topics and distribute terms among them.

Topics in LDA are defined as distributions of terms. These are sets of words that have a high probability of occurrence within a certain topic. For example, if the topic is about politics, then the most likely words in it might be "elections", "president", "parliament", etc.

Each document in the corpus is treated as a mixture of several topics. For example, an economic policy paper can be a mixture of two topics: economics and politics.

LDA assumes that the distribution of topics in each document has a predefined distribution, such as the Dirichlet distribution. This is a parameterized distribution that allows you to control how strongly one topic dominates a document. The parameter α defines this distribution of topics for documents.

$$\theta_d \sim Dirichlet(\alpha),$$

where: θ_d — topic distribution vector for document ddd, α — a hyperparameter that controls the density of topics in documents.

Each topic also has its own term distribution, which is also modeled using the Dirichlet distribution. The parameter β defines this distribution of terms for each topic.

$$\phi_k \sim Dirichlet(\beta),$$

where: ϕ_k — term distribution vector for topic k, β — hyperparameter controlling the density of terms in topics.

For each term in a document, a topic is first selected based on the distribution of topics for that document. A term is then selected based on the distribution of terms for the selected topic.

$$z_{d,n} \sim Multinomial(\theta_d),$$
$$w_{d,n} \sim Multinomial\left(\phi_{z_{d,n}}\right),$$

where: $z_{d,n}$ — topic for the nnnth term in the document d, $w_{d,n}$ — the term itself, chosen according to the topic$z_{d,n}$, θ_d — distribution of topics for the document d, $\phi_{z_{d,n}}$ — distribution of terms for the selected topic $z_{d,n}$.

The main task of LDA is to find the distribution of topics for each document and the distribution of terms for each topic. This is done using parameter estimation methods such as Expectation-Maximization (EM) or Variational Inference.

The generative LDA model is described by the following probability function:

$$P(w,z,\theta,\phi \mid \alpha,\beta) = P(\theta \mid \alpha)\prod_{k=1}^{K} P(\phi_k \mid \beta)\prod_{n=1}^{N} P(z_n \mid \theta)P(w_n \mid z_n,\phi),$$

where: w — terms in the documents, z — topics for each term, θ— distribution of topics for each document, ϕ— distribution of terms for each topic, α, β — hyperparameters of Dirichlet distributions.

3.3.7. TextRank

TextRank — is an algorithm for extracting keywords and automatically constructing text annotations, based on graph ranking methods [15]. TextRank is an adaptation of the PageRank algorithm used by search engines to rank web pages. In the case of TextRank, instead of web pages, the algorithm works on words or sentences of text.

The text is represented as a graph, where the vertices are words or sentences, and the edges are set between those vertices that are "close" in context. For keyword extraction tasks, the vertices are individual words, for summarization tasks, whole sentences.

To extract keywords:

- A term is considered a graph vertex.
- An edge connects two terms if they appear within the same window of words (usually the window has a fixed size, such as 2 or 3).

For each vertex v_i, the weight of connections with other vertices can be described through a similarity matrix, which takes into account the frequency of meeting words together in the window.

For summarization problems:

- Vertices are whole sentences.
- Edges connect sentences based on semantic similarity, which can be calculated, for example, using cosine similarity between vector representations of sentences.

The cosine similarity between sentences S_i and S_j is defined as follows:

$$cosine_{similarity(S_i,S_j)} = \frac{S_i \cdot S_j}{\|S_i\|\|S_j\|}.$$

Sentences with a high degree of similarity receive stronger connections in the graph.

The TextRank algorithm iteratively calculates the weight of each vertex (word or sentence) using the following formula, similar to the PageRank formula:

$$R(v_i\,) = (1-d) + d \sum_{v_j \in\, In(v_i\,)} \frac{R(v_j)}{(|Out(v_j)|)},$$

where: $R(v_i\,)$ — top rankv_i, d — attenuation coefficient (usually d=0.85), $In(v_i\,)$ — set of vertices that have connections withv_i, $Out(v_j)$ — the number of outgoing edges from the vertexv_j, the rank of each vertex is calculated iteratively until convergence is reached.

Keywords: After the vertices (words) are ranked, the words with the highest rank are chosen as keywords.

Text summarization: Sentences with the highest ranks are selected to construct text annotation.

3.4. Classifier

Based on the authors' previous publication [16], a comparative analysis of various machine learning methods for the classification of fake and true news was conducted. The best results were achieved using the Random Forest classifier.

Random Forest is an ensemble method that uses multiple decision trees to improve classification performance. The model consists of a set of decision trees $\{T_1, T_2, \ldots, T_N\}$, each of which is independently built on random subsets of data and features. The final classification is carried out using the voting method (choosing the class that received the most votes).

The main steps of Random Forest work:

1. Creating decision trees:

- For each tree, a subset of training data is randomly selected (with replacement, bootstrapping).
- A random subset of features is selected to construct each node of the tree.
- Tree construction continues until full classification or until the maximum depth limit is reached.

2. Voting:

- For each new example x, each tree T_i in the ensemble predicts the class y_i.
- The final prediction y is made by majority vote:

$$y = \arg\max_{y} \left(\sum_{i=1}^{N} 1\{T_i(x) = y\} \right),$$

where N is the number of trees, $T_i(x)$ is the prediction of tree T_i for the feature vector x, and $1\{T_i(x) = y\}$ is an indicator that tree T_i predicts class y.

3.5. Evaluation metrics

Metrics such as precision, recall, F1-score (F-measure) and accuracy (accuracy of classification) are used to evaluate the effectiveness of classification models. Each of these metrics has its own mathematical definition and is used to evaluate various aspects of model performance [17].

Precision shows the proportion of correct positive predictions among all positive class predictions. This is a metric responsible for the accuracy of predictions regarding positive cases (in our case, for example, fake news). Mathematically, it is represented as

$$P = \frac{TP}{TP + FP},$$

where TP — true positives, and FP — false positives.

The high accuracy indicates that the model is rarely wrong in classifying news as fake when it is true.

Recall measures the proportion of correctly predicted positive cases among all actual positive cases. This metric is important for evaluating how well the model detects fake news among all available fake news. Formula for calculation:

$$R = \frac{TP}{TP+FN},$$

where FN — false negatives.

High completeness means that the model finds the majority of all fake news, but may be wrong in classifying true news as fake.

The F1-score is a harmonic mean between precision and recall and is used to balance these two metrics, especially when it is important not only to predict the correct positive cases, but also to reduce the number of false positives and negatives. Mathematical definition of F1-score:

$$F1 = 2 \cdot \frac{P \cdot R}{P+R}.$$

This metric is particularly useful in unbalanced data situations where precision and recall may have different values.

Accuracy is an overall measure of model accuracy, showing the proportion of correct predictions among all predicted cases. It takes into account both correctly predicted positive and negative cases. Mathematically, classification accuracy is defined as follows:

$$A = \frac{TP+TN}{TP+TN+FP+FN},$$

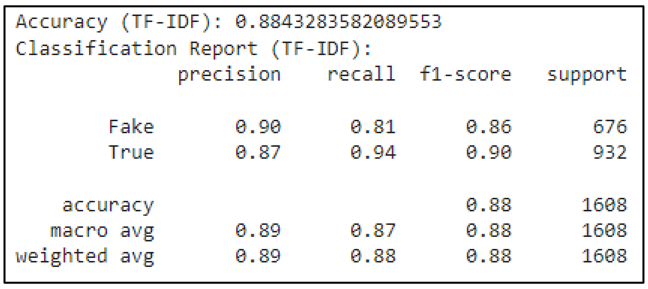
where TN — true negatives.

Accuracy is a useful metric for balanced data, but can be misleading in unbalanced class settings because it can show high accuracy even in cases where one class predominates.

4. Results

This section presents the results of a comparative analysis of the effectiveness of different methods of classifying news according to "Fake" or "True". The studied models use different keyword selection approaches, such as TF-IDF, RAKE, Yake!, LSA, LDA, and TextRank. The aim of the analysis was to evaluate the accuracy of the models using precision, recall and F1-score metrics to determine the best algorithms for detecting fake news in textual data.

Analysis of the classification report (Figure 3.) for the model based on TF-IDF shows an overall accuracy of 0.88 or 88%. For the "Fake" class, the model achieved a precision of 0.90, but a lower recall of 0.81, which indicates that the model misses some fake news. In contrast, for the "True" class, the accuracy and completeness indicators are much closer: the accuracy is 0.87 and the completeness is 0.94, which means that the model is good at recognizing true news. The final F1-score values for the "Fake" and "True" classes are 0.86 and 0.90, respectively. The macro-average and weighted average F1-score is 0.88, indicating a balanced performance of the model between classes, but with a slight advantage in detecting true news.



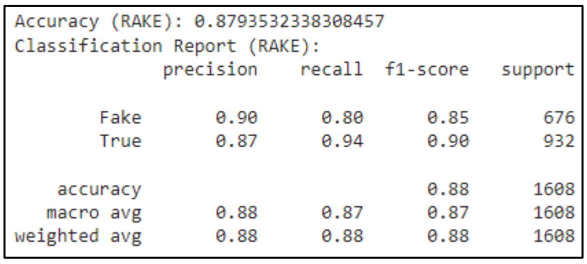
```
Accuracy (TF-IDF): 0.8843283582089553
Classification Report (TF-IDF):
              precision    recall  f1-score   support

        Fake       0.90      0.81      0.86       676
        True       0.87      0.94      0.90       932

    accuracy                           0.88      1608
   macro avg       0.89      0.87      0.88      1608
weighted avg       0.89      0.88      0.88      1608
```

Figure 3: Fake and true news classification report using TF-IDF

The classification model based on RAKE (Figure 4) showed an overall accuracy of 0.88 or 88%. For the "Fake" class, the accuracy is 0.90, but the completeness is slightly lower at 0.80, resulting in an F1-score of 0.85. The "True" class demonstrated a stable accuracy of 0.87 and a high completeness of 0.94, resulting in an F1-score of 0.90. The macro-average and weighted average value of F1-score is 0.87 and 0.88, respectively, which indicates stable performance of the model for both classes. Although the model shows balanced results, the effectiveness in detecting fake news is slightly inferior to the results for true news.



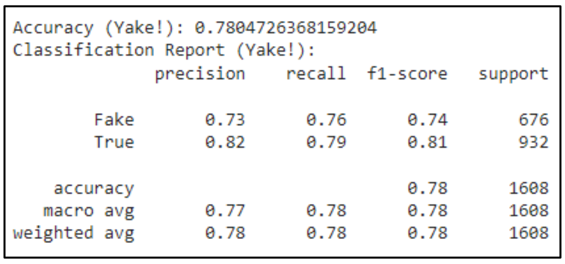
```
Accuracy (RAKE): 0.8793532338308457
Classification Report (RAKE):
              precision    recall  f1-score   support

        Fake       0.90      0.80      0.85       676
        True       0.87      0.94      0.90       932

    accuracy                           0.88      1608
   macro avg       0.88      0.87      0.87      1608
weighted avg       0.88      0.88      0.88      1608
```

Figure 4: Fake news and real news classification report using RAKE

Model Yake! (Figure 5) demonstrated an accuracy of 0.78 or 78%, which is lower compared to other models. For the "Fake" class, accuracy was 0.73, and completeness was 0.76, giving an F1-score of 0.74. For the "True" class, accuracy was slightly higher at 0.82 and completeness at 0.79, resulting in an F1-score of 0.81. The overall results show that the Yake! performs significantly worse, especially for the "Fake" class. The macro-average and weighted average value of F1-score is 0.78, which indicates a less balanced performance compared to other methods.



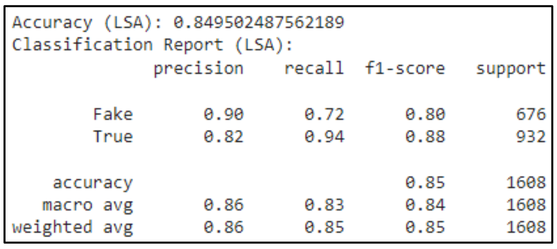
```
Accuracy (Yake!): 0.7804726368159204
Classification Report (Yake!):
              precision    recall  f1-score   support

        Fake       0.73      0.76      0.74       676
        True       0.82      0.79      0.81       932

    accuracy                           0.78      1608
   macro avg       0.77      0.78      0.78      1608
weighted avg       0.78      0.78      0.78      1608
```

Figure 5: Fake and true news classification report using Yake!

The LSA model (Figure 6) demonstrated accuracy at the level of 0.85 or 85%. For the "Fake" class, the precision was 0.90, but the completeness was 0.72, resulting in an F1-score of 0.80. For the "True" class, the indicators are significantly higher: accuracy — 0.82, completeness — 0.94, and F1-score — 0.88. Although the overall accuracy is high, the relatively low completeness for fake news suggests that the model may be missing some fake news. The macro-average F1-score is 0.84, and the weighted average is 0.85, indicating a good balance between classes.



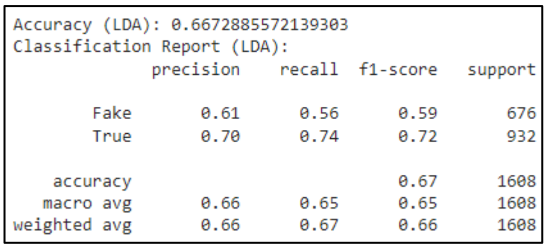
```
Accuracy (LSA): 0.849502487562189
Classification Report (LSA):
              precision    recall  f1-score   support

        Fake       0.90      0.72      0.80       676
        True       0.82      0.94      0.88       932

    accuracy                           0.85      1608
   macro avg       0.86      0.83      0.84      1608
weighted avg       0.86      0.85      0.85      1608
```

Figure 6: Fake news and real news classification report using LSA

The classification model based on LDA (Latent Dirichlet Allocation) showed (Figure 7) accuracy at the level of 0.67 or 67%, which is significantly lower compared to other models. For the "Fake" class, the precision is 0.61 and the completeness is 0.56, giving an F1-score of 0.59. For the "True" class, the accuracy is higher — 0.70, and the completeness — 0.74, which provided an F1-score at the level of 0.72. Macro-average and weighted average value of F1-score are equal to 0.65 and 0.66, respectively, which indicates an imbalance in the work of the model, in particular with a worse result for the "Fake" class. The overall results show that the LDA model has difficulty in effectively classifying fake news.



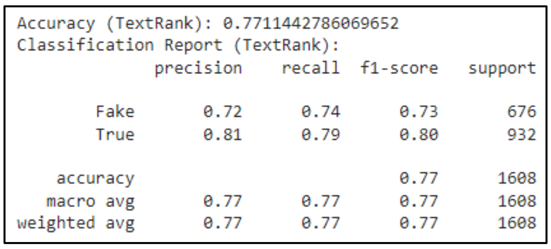
```
Accuracy (LDA): 0.6672885572139303
Classification Report (LDA):
              precision    recall  f1-score   support

        Fake       0.61      0.56      0.59       676
        True       0.70      0.74      0.72       932

    accuracy                           0.67      1608
   macro avg       0.66      0.65      0.65      1608
weighted avg       0.66      0.67      0.66      1608
```

Figure 7: Fake and true news classification report using LDA

The classification model based on TextRank (Figure 8) showed an overall accuracy of 0.77 or 77%. For the "Fake" class, the precision is 0.72 and the completeness is 0.74, giving an F1-score of 0.73. The "True" class performed better: accuracy — 0.81, completeness — 0.79, and F1-score — 0.80. Macro-average and weighted average value of F1-score are 0.77, which indicates a fairly balanced performance of the model for both classes, although the accuracy for "Fake" news is lower than for "True".


```
Accuracy (TextRank): 0.7711442786069652
Classification Report (TextRank):
              precision    recall  f1-score   support

        Fake       0.72      0.74      0.73       676
        True       0.81      0.79      0.80       932

    accuracy                           0.77      1608
   macro avg       0.77      0.77      0.77      1608
weighted avg       0.77      0.77      0.77      1608
```

Figure 8: Fake and true news classification report using TextRank

Comparing all classification methods by different metrics, it can be seen that TF-IDF and RAKE were the most effective for news classification, with accuracies of 0.8843 and 0.8794, respectively. Both methods demonstrated balanced performance for the "Fake" and "True" classes. TF-IDF has a high precision (0.90) for "Fake" news and a very high recall (0.94) for "True" news, resulting in an overall F1-score of 0.88. RAKE, although slightly inferior to TF-IDF in completeness for "Fake" news (recall 0.80), still provides a stable performance with an f1-score of 0.85 for "Fake" news and 0.90 for "True". This makes these two methods the best for detecting fake news in textual data.

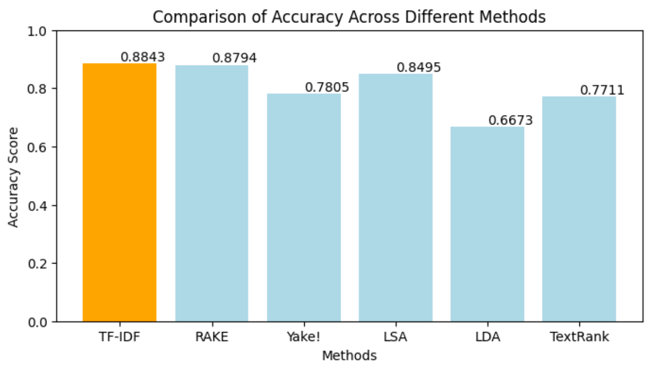


Figure 9: Comparative analysis of the accuracy of keyword selection methods for the classification of fake and true news

The results of the study showed that the TF-IDF and RAKE methods demonstrated the highest efficiency in news classification, having an accuracy of 0.88. These methods showed a balanced performance for both classes ("Fake" and "True"), especially in terms of precision and f1-score. Other methods, such as LSA, Yake!, TextRank, and LDA, performed worse, particularly in the classification of fake news, which may be due to their lower ability to accurately identify "Fake" signs. The general analysis shows that the TF-IDF and RAKE approaches are the most suitable for more accurate disinformation detection.

5. Conclusions

Within the framework of this study, a comparative analysis of keyword selection methods for classifying news into fake and true was conducted. TF-IDF, RAKE, Yake!, LSA, LDA and TextRank methods were used, which were applied to two sets of news data containing headlines from Ukrainian and Russian Telegram channels. The main objective of the study was to compare the performance of these methods using standard classification evaluation metrics such as precision, recall, F1-score and accuracy. The process included pre-processing the text, extracting keywords, building machine learning models and evaluating them based on the results obtained.

Quantitative results showed that the TF-IDF and RAKE methods became the leaders among all the tested approaches. TF-IDF demonstrated the highest precision of 0.8843, with a high precision for fake news (0.90) and a very high recall for true news (0.94), providing an F1-score of 0.88. The

RAKE method performed slightly lower, with an overall accuracy of 0.8794 and an F1-score of 0.85 for fake news and 0.90 for true news. Other methods such as LSA (0.8495), Yake! (0.7805), TextRank (0.7711) and LDA (0.6673), showed worse results. Particularly low results were obtained in the classification of fake news, which indicates the difficulty of these approaches in identifying false information. This approach can be used to detect fake information, in particular, during an information war.

In the future, it is planned to develop a tool for detecting fake news, which will be based on the results of this study. The tool will integrate the most effective keyword selection methods, including TF-IDF and RAKE, to analyze textual data from news sources. The main goal will be to create a system capable of quickly and accurately identifying fake information based on the latest machine learning methods.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] Khaleel Ur et al., Comparative study of fake news detection between machine learning and deep learning approaches. In: Proceedings of the 1st National Conference on Applications of Soft Computing Techniques in Engineering NCASCTE-2022, Hyderabad, India, 2023, pp. 50-56. https://www.doi.org/10.56726/IRJMETS-NCASCTE202230

[2] T. Chauhan, H. Palivela, Optimization and improvement of fake news detection using deep learning approaches for societal benefit. International Journal of Information Management Data Insights 1 (2021) 100051. https://doi.org/10.1016/j.jjimei.2021.100051

[3] E. A. Johnson et al., An experimental comparison of classification tools for fake news detection. International Journal of Advanced Research in Computer and Communication Engineering 10 (2021) 135-141. https://doi.org/10.17148/IJARCCE.2021.10820.

[4] S. Garg, D. K. Sharma. Linguistic features based framework for automatic fake news detection. Computers & Industrial Engineering 172 (2022) 108432. https://doi.org/10.1016/j.cie.2022.108432

[5] M. C. de Souza, M. P. S. Gôlo, A. M. G. Jorge, E. C. F. de Amorim, R. N. T. Campos, R. M. Marcacini, S. O. Rezende. Keywords attention for fake news detection using few positive labels. Information Sciences 663 (2024) 120300. https://doi.org/10.1016/j.ins.2024.120300

[6] Ukrainian news. URL: https://www.kaggle.com/datasets/zepopo/ukrainian-fake-and-true-news

[7] Sophia Matskovych. Fake News UA. URL: https://www.kaggle.com/datasets/sophiamatskovych/fake-news-ua

[8] A. Aizawa. An information-theoretic perspective of TF–IDF measures. Information Processing & Management 39 (2003) 45-65. https://doi.org/10.1016/S0306-4573(02)00021-3

[9] S. Rose, D. Engel, N. Cramer, W. Cowley. Automatic keyword extraction from individual documents. Text mining: applications and theory, Editor(s): Michael W. Berry, Jacob Kogan (2010) 1-20. https://doi.org/10.1002/9780470689646.ch1

[10] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, & A. Jatowt. YAKE! Keyword extraction from single documents using multiple local features. Information Sciences 509 (2020) 257-289. https://doi.org/10.1016/j.ins.2019.09.013

[11] P. Pęzik, A. Mikołajczyk, A. Wawrzyński, B. Nitoń, M. Ogrodniczuk. Keyword extraction from short texts with a text-to-text transfer transformer. In Asian Conference on Intelligent Information and Database Systems. Singapore: Springer Nature Singapore. 2022, pp. 530-542. https://doi.org/10.1007/978-981-19-8234-7_41

[12] T. K. Landauer, P. W. Foltz, D. Laham. An introduction to latent semantic analysis. Discourse processes 25 (1998) 259-284. https://doi.org/10.1080/01638539809545028

[13] K. Lipianina-Honcharenko, T. Lendiuk, A. Sachenko, O. Osolinskyi, D. Zahorodnia, M. Komar. An intelligent method for forming the advertising content of higher education institutions based on semantic analysis. In: International Conference on Information and Communication Technologies in Education, Research, and Industrial Applications. Cham: Springer International Publishing. September 2021, pp. 169-182. https://doi.org/10.1007/978-3-031-14841-5_11

[14] D. M. Blei, A. Y. Ng, M. I. Jordan. Latent Dirichlet allocation. Journal of Machine Learning Research 3 (2003) 993-1022.
[15] M. Zhang, X. Li, S. Yue, L. Yang. An empirical study of TextRank for keyword extraction. IEEE Access 8 (2020) 178849-178858. https://doi.org/10.1109/ACCESS.2020.3027567
[16] K. Lipianina-Honcharenko, M. Soia, K. Yurkiv, A. Ivasechko. Evaluation of the effectiveness of machine learning methods for detecting disinformation in Ukrainian text data. In: Proceedings of the Seventh International Workshop on Computer Modeling and Intelligent Systems (CMIS-2024), Zaporizhzhia, Ukraine, May 3, 2024. https://ceur-ws.org/Vol-3702/paper9.pdf
[17] K. Lipianina-Honcharenko, C. Wolff, A. Sachenko, I. Kit, D. Zahorodnia. Intelligent method for classifying the level of anthropogenic disasters. Big Data and Cognitive Computing 7 (2023) 157. https://doi.org/10.3390/bdcc7030157