Information system for collaborationists messages and accounts classification in social networks

Victoria Vysotska[1,†], Viktoriia Yakovlieva[1,†], Roman Romanchuk[1,*,†], Lyubomyr Chyrun[1,2,†], Dmytro Uhryn[3,†], Vitalii Danylyk[1,†], Oksana Brodyak[1,†], Ainur Orazayeva[4,†] and Vadim Schuchmann[5,†]

[1] *Lviv Polytechnic National University, S. Bandera Street 12, 79013 Lviv, Ukraine*

[2] *Ivan Franko National University, Universytetska Street, 1, 79000 Lviv, Ukraine*

[3] *Yuriy Fedlovyvh Chernivtsi National University, Kotsiubynskoho Street 2, 58012 Chernivtsi, Ukraine*

[4] *Kazakh University of Technology and Business named after K. Kulazhanov, Left Bank, Nura district, st. Kaiym Mukhamedkhanova, building 37 A, 010000 Astana, The Republic of Kazakhstan*

[5] *West Ukrainian National University, Lvivska Street 11, 46004 Ternopil, Ukraine*

Abstract

The article presents the results of the first comprehensive study in Ukraine dedicated to the automatic detection of collaborationist messages and the classification of user accounts in social networks. Based on a unique, hand-labelled dataset that includes more than 12,862 texts and extended corpora of over 140,000 and 400,000 messages from various Telegram groups, two models have been developed: a multi-class model for classifying messages (pro-Ukrainian, neutral, collaborationist) and a binary model for classifying accounts. To improve accuracy, an ensemble approach using TF-IDF, symbolic n-grams, naïve Bayesian and logistic regression was applied. The developed system demonstrates high accuracy on test data (Accuracy = 0.9438) and is capable of effectively analysing large arrays of texts in batch processing mode. The paper conducts a comparative analysis with modern research in the field of disinformation detection, identifies the key advantages of the approach, and formulates the practical value of the system for law enforcement agencies, think tanks, and information security projects. The presented results provide the basis for further development of tools for the automated detection of collaborators in the digital space.

Keywords

Collaborationism, machine learning, NLP; text classification, Telegram, information security, propaganda detection, Ukrainian information space, ensemble models, text vectorisation, TF-IDF, n-grams, account classification, social media analysis.

1. Introduction

On social networks, you can find many different opinions on a variety of topics. In such an environment, people feel more relaxed and at ease, allowing them to express their thoughts without hesitation. Therefore, social networks are a reliable source for analysing the genuine opinions of society. This effect of social media fluency is amplified when people write messages in a group of like-minded people. That is why collaborative groups on social networks are the best environment for the spread of traitors to Ukraine. People from the occupied territories are generally divided into three groups: those who left the occupation, those who remained for their own reasons but do not hold pro-Russian views, and those who have been waiting for Russia for a

AIT&AIS'2025: International Scientific Workshop on Applied Information Technologies and Artificial Intelligence Systems, December 18–19 2025, Chernivtsi, Ukraine

[1,*] Corresponding author.

[†] These authors contributed equally.

victoria.a.vysotska@lpnu.ua (V. Vysotska); viktoriia.yakovlieva.sa.2022@lpnu.ua (V. Yakovlieva); roman.v.romanchuk@lpnu.ua (R. Romanchuk); lyubomyr.v.chyrun@lpnu.ua (L. Chyrun); d.ugryn@chnu.edu.ua (D. Uhryn); vitalii.m.danylyk@lpnu.ua (V. Danylyk); brodyakoksana1976@gmail.com (O. Brodyak); oaris.83@gmail.com (A. Orazayeva); V.shukhmann@st.wunu.edu.ua (V. Schuchmann)

0000-0001-6417-3689 (V. Vysotsk); 0009-0003-2887-173X (V. Yakovlieva); 0009-0004-4352-1073 (R. Romanchuk); 0000-0002-9448-1751 (L. Chyrun); 0000-0003-4858-4511 (D. Uhryn); 0000-0001-5928-7235 (V. Danylyk); 0000-0002-9886-3589 (O. Brodyak); 0000-0002-2899-9886 (A. Orazayeva); 0000-0002-1427-3312 (V. Schuchmann)

© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

published 2026-02-07

CEUR-WS.org/Vol-4160/paper12.pdf

long time and are now enjoying its occupation. People from the occupied territories are concentrated in local groups of a specific city or town, and these are usually two types of groups: pro-Ukrainian, waiting for the return of Ukraine, there are people from the first two groups; and pro-Russian groups, where individuals from the last group are concentrated, they rejoice at Russia and wait for it to seize more territories. Information will be collected from the previous types of groups. This topic of searching for and condemning collaborators is currently very relevant, since these persons in the occupied territories are extremely dangerous. The main reason is that they inform the Russian military about the exact whereabouts of people who had or still have a pro-Ukrainian position, former Ukrainian soldiers, or even their relatives. What happens to our people afterwards is exceedingly terrible, to put it briefly, and in veiled words used by people from the occupied territories, they are "put in basements". These persons may still not be punished, even after the de-occupation of Ukrainian territories, which makes our people feel unnecessary and unprotected, as these collaborators who "put them in basements" are now calmly walking around the cities. That is why we need to support the law enforcement system and ultimately convict collaborators in accordance with Ukrainian law. Currently, there is no such system in Ukraine, and no dataset exists on this topic. The only publicly available information is the existence of websites and Telegram groups that list collaborators, most of whom are from specific regions, along with a record of crimes committed by them.

The purpose of this study is to develop an information system for searching and classifying collaborative messages, as well as analysing and categorising user accounts. Main tasks:

1. Create a system that classifies users' messages on social networks as pro-Ukrainian/neutral/collaborationist: Create a dataset of 10862 lines (messages), that is, extract messages from a certain collaborator from social networks; Manually mark the dataset for 10862 messages; Create a dataset from messages taken from several collaborationist groups.
2. Divide the dataset from different users into two parts: one containing users who wrote fewer than 50 messages, and the other containing the rest. The part of the dataset containing messages from users who have written fewer than 50 messages will be used for this model, while the other part will be used for the next model.
3. Create a dataset based on a pro-Ukrainian chat, where all pro-Russian messages are deleted, and mark all messages as (-1).
4. Manually mark 2032 messages from the dataset, where the messages of users who have written fewer than 50 messages.
5. Leave 2,000 lines in the dataset with pro-Ukrainian messages to prevent class imbalance.
6. Train the initial model on a dataset of 10,862 and 2,000 pro-Ukrainian messages, as well as 2,032 pro-Russian messages.
7. Label 88864 messages based on the created model.
8. Find a model that would allow you to most accurately mark the rest of the messages automatically based on 11000 messages from one collaborator and part of the checked and relabeled messages from different users.
9. Create an ensemble of the best models on 141357 messages.
10. Train the ensemble.
11. Mark the rest of the messages from different users based on this ensemble.
12. Create an ensemble based on the best models.
13. To teach this ensemble.
14. Test the model.
15. Create a system that classifies users' social media accounts as collaborators/non-collaborators: Based on the last ensemble, mark the dataset of messages for which users who wrote more than 50 messages wrote.
16. Find the conditions under which accounts will be classified as collaborationist.
17. Mark the accounts of users who wrote messages in the pro-Ukrainian group as 0.

18. Find the models that show the best results.
19. Train the model.
20. Test the model.

The object of the study is the processes of analysing information messages spread in social networks. The subject of the study is the methods and means of searching, analysing, and classifying messages of a collaborationist nature, based on the principles of computational linguistics and machine learning. Currently, no publicly available information system classifies collaborationist messages in Ukraine. There is no such dataset, so I had to create and label it myself. If you look for systems with similar topics, then the closest thing you can find is systems that detect false or spammy content, rumours or crises. Such systems utilise NLP, ML, VAE, belief functions, and a combination of human and ML [1–4]. That is, computational linguistics and machine learning already have examples of models that will be used in this study; however, they have never been applied in such a context, for such a purpose, at least in the public domain. Such a system cannot be found.

This system can help law enforcement in Ukraine identify traitors to Ukraine (collaborators), as manually checking the social networks of each person is an impossible task in the modern world. This system will help you quickly check messages written by any person and determine if there are collaborative sentiments in their messages. Thus, law enforcement officers will be able to promptly identify traitors and open cases against them, and eventually convict offenders. Additionally, this system can be applied directly to social networks to detect and remove dangerous messages. Thanks to this, Ukrainians will no longer need to see Russian propaganda on their social networks. As a result, this will lead to a decrease in the number of collaborators themselves, since the fewer people who read Russian propaganda, the fewer people who believe in it. In addition, it would be possible to create a startup based on this system that would find pro-Russian messages and hide them from the user, similar to such systems as stopRU (a browser extension that blocks Russian videos on YouTube), RuFilter (an extension that allows you to disable .ru domains, ru.wikipedia.org, sites with Russian letters) [5–6]. It is also possible to create a similar system to identify collaborators in countries where there is a serious threat of information occupation, such as the Baltic countries, Georgia, and Poland. With the help of such systems, it would be possible to filter pro-Russian messages in these countries as well, because there are collaborators not only in Ukraine but also in Russia, who do not plan to stop at us; our partners must also see and distinguish the threat [7–10].

2. Related works

Currently, two models that were created during the study (a model that classifies messages as pro-Ukrainian/neutral/collaborationist, and a model that classifies accounts as neutral and collaborationist) are the first such models due to their unique themes. The closest analogues that can be found for the first model in classifying collaborationist messages are the XLNet model [11], which categorises pro-Russian/neutral/pro-Ukrainian tweets, and a study [12–24]. Next, we will carry out a detailed analysis and describe each such study.

In [11], tweets from pro-Russian, neutral, and pro-Ukrainian sources were used for this model. The results showed that accuracy initially improved with increasing data volume, but decreased slightly when multiple datasets were used, indicating the need for a balance between data quality and quantity. This research enhances the ability to detect and analyse disinformation in real-time, thereby contributing to effective public information management and strategic communication efforts during war. Despite the use of advanced models, research faces limitations such as dataset bias, generalisation problems, high computational requirements, and a focus on specific platforms, which limit the widespread application of their findings. XLNet is an advanced NLP model designed to overcome the limitations of previous models, such as BERT. It is based on the Transformer architecture, but XLNet is distinguished by its new approach to permutation-based

learning. Unlike BERT, which employs a masked language modelling technique, XLNet utilises an autoregressive permutation-based learning objective, enabling the model to efficiently capture bidirectional context while preserving natural language structure. This permutation approach allows the model to learn the relationships between words in a more general manner, thereby enhancing its understanding of linguistic relationships within a sequence. Unlike BERT, which relies on masked language modelling, XLNet generates predictions by maximising the probability of a sequence under all possible permutations of the factorisation order [11–12].

Article [13] is a study that analyses how the media influenced and reflected public opinion during the first month of the war, utilising news and Telegram channels in Ukrainian, Russian, Romanian, French, and English. They implemented binary classification using the following models for input vectors consisting of 41 self-generated linguistic features and 116 keywords (normalised by text length in tokens): decision tree, linear regression, support vector machine (SVM), and neural networks, using stratified 5x cross-validation (10% for testing and 90% for training). For comparison with the learned features, they extracted an embedding using the multilingual BERT model and trained the linear model using it. These researchers collected 18,229 texts, 8,872 of which contained pro-Western narratives and 9,357 with pro-Russian narratives, and conducted three experiments: the first used data only from the news, the second used data only from Telegram, and the third used a mixed set. Additionally, the researchers analysed their best-performing SVM model to determine the importance of both linguistic traits and keywords using the trait permutation method. Disadvantages of this study include poor generalisation to Telegram and French and Romanian texts, as well as limited analysis of multimodal signals (images and videos). The figure shows how the researchers identified the advantages and disadvantages of the methods they used:

The article [14] describes the process of creating a dataset comprising over 38 million posts from Russian media, including state-owned outlets, as well as from social networks such as VK and Twitter. VK primarily presents media to an audience within the country, while Twitter presents it to an audience outside. Two different thematic models were employed: a structured topic model and a contextually informed neural topic model (CTM - Correlated Topic Model). The Structural Topic Model (STM) is a popular LDA-style probabilistic model that improves on previous approaches by allowing users to include arbitrary metadata. CTM is based on a variational autoencoder and incorporates pre-trained sentence embeddings into document representations, thereby reducing the assumptions made about the word set by traditional models. The main results obtained by the researchers are as follows: state media paid more attention to military topics.

In contrast, independent media used the term "war" more often, whereas state media used the term "operation". Scientists also encountered three main limitations: in interpretation, instability, and oversimplification. In data, not all topics are consistent, and even in sequential issues, it is challenging to determine completeness. For example, the CTM topic most relevant to Ukraine (Topic 5) refers to two mostly unrecognised breakaway states in eastern Ukraine: the DPR and the LPR, suggesting that this topic reflects clearly pro-Russian coverage of events, which we expect to be more prevalent in state media. While word statistics and manual analysis suggest that independent media discuss the war more frequently than state media, the model's theme models indicate otherwise. Volatility makes it challenging to trust the results, and more research is needed to improve consistency and reliability. Word-level metrics do not take into account context, and most thematic models, including STMs, make assumptions about "word bag" and independence. While CTM weakens the assumption of a "word bag" by embedding sentences, the disadvantage compared to STM is that it does not parameterise topics using metadata.

The article [15] examines the application of GNN in detecting fake news, particularly in the context of disinformation campaigns, such as those observed in Russia's information war against Ukraine. This study focuses on automating the analysis of negative psychological impact in online media using knowledge graphs (KG) and GNN-based models, including GraphSAGE, GAT and GCN. By encoding relationships in knowledge graphs, these techniques facilitate the detection of harmful content shared on social media. Despite promising results, the study highlights several limitations, including reliance on large, labelled datasets, issues with the stability and accuracy of

models across different platforms, and the need for substantial computing resources, particularly for real-time monitoring.

The article [16] explores how social media users, in particular on the Russian subreddit of the Reddit platform, act as "visual gatekeepers of the audience", selectively sharing their view of the world in order to influence public opinion, especially during Russia's war against Ukraine. Through critical analysis of visual content, the study examines how goalkeepers create a visual "information bilbashka" that reinforces their social reality and ideological views. The main findings reveal that during polarising events, users reinforce specific narratives, often portraying Russia in a favourable light and condemning perceived opponents, which can lead to the radicalisation and bias of visual content. A limitation of this research is its focus on a single subreddit, which may not reflect broader audience dynamics on other platforms or in different contexts.

The article [17] examines the themes and sentiments expressed by Ukrainian-speaking Telegram users during the first six months of Russia's war against Ukraine, using machine learning techniques to analyse social media data. This study employs theme modelling using non-negative matrix factorisation with Kullback-Leibler divergence and sentiment analysis utilising pre-trained models to categorise themes and emotional colouring of messages. A significant limitation is the concentration of the data set on a single platform (Telegram), which may not cover the entire spectrum of social discussions around the war.

The article [18] analyses Russia's bot-based propaganda and Ukraine's social media counternarratives during key stages of Russia's war against Ukraine in 2022. Using TweetBERT to model topics and integrating the BEND framework with Moral Foundations Theory, this study examines how bots manipulated narratives to justify Russia's actions and counter NATO, while Ukraine used similar tactics to promote solidarity and resilience. The main limitation of this study is its exclusive focus on bot-generated content, which may not fully reflect human interactions or the overall impact on public opinion.

The article [19] examines the impact of Twitter's policy on labelling Russian state media accounts, analysing whether labelling has reduced the reach and influence of these accounts since the start of Russia's war against Ukraine. Using the ARIMA model to track engagement rates before and after the implementation of Twitter labelling on February 28, this study measures changes in tweet reach by focusing on the number of retweets. The main limitation is the lack of a causal relationship between Twitter's labelling policy and the decline in engagement, which may be attributed to concurrent events such as the restriction of Russian media in Europe and the blocking of Twitter by Russia.

The article [20] presents the OLTW-TEC method, an advanced machine learning approach for detecting disinformation in Ukrainian-language materials that utilises a set of text classifiers and a sliding window for dynamic online learning. The proposed method combines several classifiers to adapt to changes in data, ensuring high accuracy and relevance in real-time scenarios. A significant limitation of the proposed method is the high computational requirements, which can hinder scalability, especially for large-scale or resource-constrained applications.

The article [21] examines the processing of Russian disinformation about the war using three popular LLM-based chatbots: Perplexity, Google Bard, and Bing Chat. Using an AI audit approach, this study examines the consistency, accuracy, and use of disclaimers in chatbot responses to queries related to Russian disinformation narratives. The main limitation of this study is the LLM's inherent stochasticity, which leads to significant variability in results and often inadvertently reinforces false narratives.

The article [22] presents the Entity-Aware Approach (EAA) to identify logical errors in Kremlin-related social media content, specifically targeting disinformation about the war in Ukraine. Using Named Object Recognition (NER), EAA replaces named objects with generic labels to improve model performance by reducing confusion in error detection, especially when applied to Kremlin tweets. The results demonstrate that when combined with the DeBERTa language model, EAA outperforms the underlying models on both non-domain-specific datasets (LOGIC) and domain-specific datasets (RuFal). However, the current study is limited in that it relies on a single

NER approach and dataset. Therefore, additional datasets and ensemble methods should be considered in future work.

The article [23] explores the role of Ukrainian and Russian diaspora communities in spreading disinformation on social media, with particular emphasis on content related to the war in Donbas and the MH17 crash. This study uses a combination of social media analysis and ML classification methods to identify user communities and classify them by diaspora affiliation (Ukrainian, Russian, or other). A significant limitation of the study is the lack of multilingual data, as it is limited to English-language tweets, which may not accurately reflect the full extent of diaspora participation in disinformation campaigns.

The purpose of the latest study [24] is to show the potential of social networks for cyber intelligence in the context of the Russian-Ukrainian cyber war. Language detection and translation, sentiment analysis, TF-IDF, LDA, Porter stemming, n-grams, and real-time monitoring using the Twitter API were used for analysis. The results of the study showed that discussions about sanctions and energy security dominate the topics of Russian propaganda, the peak of negative sentiments fell on mass DDoS attacks at the end of November 2022, and a high awareness of Ukrainians on the topic of information security and "cyber patriotism" was also recorded on the positive side. Disadvantages of the study include multilingualism and translation, data noise, short texts, API limitations, topic shift, class balance, credibility, multimodality, georeferencing, and distribution of attacks.

Let's compare the results of the research [11–24] with those of our study. In contrast to the XLNet-based research [11], we used a variety of data sources to identify collaborationism: from collaborationist groups on Telegram to a highly patriotic group on Telegram (in particular, Sternenko's chat [24]). In addition, only 42000 data points were used in this study. In contrast, 103,759 messages were used for the classification model of collaborationist messages (10,862 from one collaborator, 2,000 messages from a pro-Ukrainian group, and 90,897 messages from a collaborationist group). Additionally, the researchers manually labelled only 5,000 messages, while 12,862 messages were manually labelled.

Although the study [14] has a sufficient amount of data (38 million), which is more than my dataset, and examines messages from two social networks – VK and Twitter, the researchers only created a dataset, not a machine learning module. This dataset contains only pro-Russian messages, which will complicate future research based on it, as it requires pro-Ukrainian and neutral data to train quality and impartial models. Although the study [15] bears almost the greatest similarity to mine, ours has significant advantages and differences. In particular, the researchers limited themselves to only 18229 messages. In comparison, ours includes 401601 messages for the model that classifies user accounts (372941 messages from pro-Russian groups and 39601 from pro-Ukrainian groups) and 141360 messages for the model that examines messages, without attribution (39601 of the same data from a pro-Ukrainian chat, 90897 messages from pro-Russian chats, and 10862 manually labelled messages from an individual collaborator. Compared to my metrics, their models yield mediocre results, despite using more modern models.

In contrast to the study [15], our models yield excellent results on various platforms. Notably, we tested messages from accounts extracted from Twitter, and the model demonstrated quite good results. Most likely, the size of my dataset for the model affected its performance, allowing it to distinguish between different messages. Unlike the study [16], which investigated only an "information bubble" in the form of a Russian subreddit, models about collaborators were trained on data from two pro-Russian and one pro-Ukrainian Telegram chats.

The advantage of our models compared to the study [17] is that the data used to train them includes not only Ukrainian-speaking users but also Ukrainian speakers, Russian speakers, and users with a mixture of these languages. The study [18] only examined messages written by bots, whereas our research utilised messages written by real people. In addition to the fact that the study [19] used data from Twitter, it examined only Russian media. It identified the lack of a causal relationship between Twitter's labelling policy and the decline in engagement, which was

attributed to concurrent events such as the restriction of Russian media in Europe and the blocking of Twitter by Russia.

The primary issue with the study [21] is the inherent stochasticity of the results, which leads to significant variability in responses and often inadvertently reinforces false narratives. In contrast, naïve Bayesian and logistic regression models do not have such problems. In addition, this study focused solely on chatbots, whereas our message included collaborators, which changes the study's purpose. There is a significant drawback in the study [22], as it relies on a single NER approach and dataset. In contrast, different datasets are used, and an ensemble is created by combining the two vectorisation methods and models. In addition to the fact that the study [23] focuses solely on diasporas, a disadvantage of this study is the lack of multilingual data, as it is limited to English-language tweets, which may not accurately reflect the full extent of diaspora participation in disinformation campaigns.

Although many have researched Russian propaganda and Russian narratives [1–10], no one has yet investigated the role of collaborators. And their rhetoric is still different, for example, he writes: "I'm glad the occupation came, when Russia came, it got better," while the propagandist writes: "The Ukrainian military is shelling themselves," although the collaborator uses the rhetoric of a propagandist in many ways, he still has unique types of messages that are not characteristic of others. Main disadvantages:

1. One social network.
2. Restrictions by polar groups.
3. Marking all messages from the pro-Ukrainian group as pro-Ukrainian.
4. Language restriction.

To train the model, data was used exclusively from Telegram, ignoring other social networks, such as Twitter, Instagram, or Vkontakte. It can lead to the fact that models can understand and classify messages written on the Telegram social network well, but show inferior results when given messages written on other social networks, since the structures of writing messages on Telegram and other social networks differ.

Only two polar groups on Telegram were used to train and test the models: two collaborationist groups and one pro-Ukrainian group. It can lead to models poorly targeting "neutral people", i.e., those who do not express a pro-Ukrainian or pro-Russian position. Although models know what neutral messages look like, they are often presented by people who already have a clear pro-Ukrainian or pro-Russian stance.

To position users with a pro-Ukrainian position, not counting users in pro-Russian groups that acted as the opposition, messages from a pro-Ukrainian group were used, where all Russian propaganda is filtered (a check was carried out, and no pro-Russian messages were found), so it was decided to mark all messages from this group as pro-Ukrainian messages. It can lead to the model recognising neutral messages as pro-Ukrainian.

Only two languages were used for the models: Ukrainian and Russian, as well as a combination of these languages. It entails significant limitations; in particular, the model cannot recognise other languages, although collaborators can write messages in different languages, spreading propaganda to users who do not know Ukrainian and Russian.

The primary issue is that there is no single model that examines collaborators, although numerous studies have been conducted on Russian propaganda. Additionally, in studies examining Russian propaganda, there is typically very little data available, so it is crucial to utilise sufficient data. It is necessary to develop the first model in Ukraine that classifies collaborationist messages, as well as a model that categorises accounts as neutral or collaborationist.

3. System analysis of the product development

The presented work is devoted to the development of an automated information system, "Sphere1", designed for monitoring, collecting, analysing, and classifying information flows (text messages) in social networks. The primary goal of the project is to automatically identify and categorise content and accounts that exhibit signs of collaborationist activity. This toolkit is critical for ensuring information security and countering disinformation campaigns. The Sphere system is aimed at a wide range of institutions and researchers whose activities are related to the analysis of the information space and national security:

1. Moderators and Analysts of Information Security in Government Institutions.
2. Law enforcement agencies are responsible for identifying and documenting threatening information activities.
3. Media and investigative journalists who monitor the information environment.
4. Researchers in propaganda and disinformation use techniques to obtain large, labelled data sets.

In the context of this study, the following specialised terms are used, presented in Table 1.

Table 1
Key concepts and terminology

Term	Definition
Stance detection	Automated categorisation of the text based on the author's attitude to a specific topic. In this system, classification is carried out into three categories: pro-Ukrainian, neutral, and collaborationist.
Collaborative content	Messages or patterns of account behaviour that express support for the occupation administration or spread pro-Russian propaganda rhetoric.
Weighted count vectorizer	A method of constructing a vector representation of text, where each lexical element (token) is assigned a weight reflecting its importance in the corpus (for example, TF-IDF).
Count character *n*-grams	Vectorisation of text based on the frequency of occurrence of sequences of characters of a given length. It is used to increase the model's resistance to spelling, slang, and transliteration variations.
Naïve strict	Specific configuration of a naïve Bayesian classifier (in particular, Multinomial Naive Bayes) with tight settings of a priori probabilities.
Logistic regression	Linear model used for binary or multiclass classification. Evaluates the probability of an object belonging to a particular class through the logistics function.

The analysis of the current situation in the field of detecting hostile information activities revealed a number of methodological and empirical gaps that need to be addressed:

1. Research focus.
2. Limited training data.
3. Ethical challenges and classification accuracy.
4. Class Imbalance and the Problem of Pro-Ukrainian Messages.

The vast majority of scholarly papers focus on general aspects of Russian propaganda and pro-Russian narratives. There is a shortage of specialised studies that focus directly on identifying collaborationist content. Current models are often trained on relatively small datasets (up to 20,000 examples), and the number of qualitatively manually labelled messages usually does not exceed 5,000. It significantly limits the generalisation and accuracy of existing classifiers. The use of classifiers at the state level requires extremely high precision. False positives can have serious social consequences. Existing systems, which demonstrate a metric $F_1 \leq 0{,}93$, even on larger samples, still carry significant risks. Most models do not adequately account for or handle pro-Ukrainian messages, which often leads to their misclassification as "pro-Russian", likely due to similarities in the level of aggressive language. The introduction of a separate label for pro-Ukrainian messages is crucial to enhancing the model's accuracy.

The Sphere system is designed as a two-component tool consisting of a model training module (Table 2) and a new data classification module (Table 3). The system is an ensemble of machine learning models designed for automatic multiclass classification of Ukrainian text messages, taking into account technical limitations (Table 4). Main goals:

1. Automatic classification of texts into three target categories: pro-Ukrainian, neutral, and collaborationist.
2. Achieving high classification accuracy through the use of an ensemble architecture.
3. Processing large amounts of data (up to 400,000+ messages).
4. Providing a Confidence Score for each forecast.

Table 2
Model training module detail

Function	Purpose	Job Description	Input/Output
load_and_ preprocess _datasets	Loading, processing and unifying multiple training datasets.	1. Upload files A, B, C (with different column names). 2. Standardisation of columns to ["text", "label"]. 3. Combining into a single DataFrame. 4. Clearing of empty/incorrect entries. 5. Application of pre-treatment.	Output: combined, cleaned dataset.
preprocess _text	Standardisation and purification of a single text string.	1. Processing of NaN values. 2. Conversion to lowercase. 3. Removal of extra characters/spaces. 4. Normalisation of spaces.	Input: text string. Output: cleaned, normalised text.
reate_ense mble_ model	Initialisation of ensemble components (vectorizers and classifiers).	1. Creating a TfidfVectorizer (max_features=10000, n-gram=(1, 2)). 2. Creating a CountVectorizer for the character level (analyzer='char', n-gram=(2, 4), max_features=5000). 3. Initialisation of MultinomialNB (alpha=1.0) and LogisticRegression (with optimisation parameters).	Output: A set of non-configurable vectorizers and models.
train_ ensemble	A complete cycle of ensemble training and quality	1. Label coding, data separation (80% Train, 20% Test) with stratification. 2. Vectorisation of texts in two ways (TF-IDF and symbolic). 3. Training of individual models. 4. Creating an	Input: array of texts X_y, variety of labels. Output: trained components, quality metrics.

	assessment.	ensemble through averaging probabilities. 5. Calculation of quality metrics (accuracy, F1, classification report).	
save_ model	Serialisation of the trained model for permanent storage.	Save the complete component dictionary (vectorizers, models, label encoder, metrics) to a .pkl file using joblib.	Output: Model file on disk.
predict_ with_ ensemble	Classification of a single text using an ensemble.	1. Pre-processing of the input text. 2. Double vectorisation. 3. Getting probabilities from each model. 4. Averaging probabilities for the final forecast. 5. Decoding the label and calculating the confidence level.	Input: text, model components. Output: predicted label, confidence level.

Table 3
Detailing the new data classification module (batch processing)

Function (Program Name)	Purpose	Job Description	Input/output
load_model	Deserialization of a saved model from disk.	Loading a dictionary of components from a .pkl file using joblib, checking for the existence of the file and handling errors.	Input: The path to the model file. Output: Model Component Dictionary or None.
predict_batch	Efficient batch forecasting for large amounts of data.	1. Splitting input data into packets (default is 1000). 2. Pre-processing and filtering of blank texts. 3. Parallel vectorisation and forecasting to improve efficiency. 4. Progress Display (tqdm).	Input: list of texts, model, packet size. Exit: lists of predicted labels and confidence.
load_and_ predict_ dataset	A complete cycle of labelling a large dataset (for example, 400,000 messages).	1. Uploading an Excel file, validating the structure. 2. Call batch prediction for all texts. 3. Adding results to the DataFrame. 4. Saving the results to a new Excel file.	Input includes the input file, model, and save path. Output is a DataFrame with forecasts.
analyze_ predictions	Detailed statistical and qualitative analysis of classification results.	1. Categorisation of results by confidence levels (Low: 0.0-0.5, Medium: 0.5-0.7, High: 0.7-0.9, Very High: 0.9-1.0). 2. Statistical analysis of the distribution. 3. Demonstration of examples of the most/least reliable forecasts.	Input: DataFrame with forecasts. Output: text analytical report.

Table 4
Technical limitations

Category	Requirement/Restrictions
Hardware requirements	RAM minimum 8 GB for processing large datasets (400,000+). HDD with a minimum of 2 GB to store models and results. CPU – Recommended multi-core processor.
Software dependencies	Python 3.7. Required libraries: pandas, numpy, scikit-learn, joblib, tqdm, openpyxl (with eligible versions).
Data Limitations	Format Excel files only (.xlsx). UTF-8 encoding. Required columns: "message text" (for classification), "text" and "label" (for learning). The text length is optimally up to 1000 characters.
Functional limitations	Supported Operations: Multiclass Classification, Confidence Score, and Batch Processing. Does not support regression, clustering, or online learning.
Model limitations	The type of training is exclusively Supervised Learning. Architecture – fixed ensemble (Naive Bayes and Logistic Regression). Needs complete retraining.

Performance limitations and accuracy:

1. Processing speed, in particular, involves training for 10-30 minutes on datasets of up to 10,000 records, classification (batch) of approximately 1000-2000 texts per minute (depending on equipment configuration), and a recommended batch size of roughly 500-2000 records.
2. Memory limit, i.e. the maximum number of characters, is fixed (10,000 TF-IDF + 5,000 characters) and recommended maximum dataset size: up to 500,000 records.
3. Expected accuracy, in particular, optimal conditions: 80-95% accuracy, real conditions (non-ideal data): 70-85% accuracy and minimum acceptable accuracy 60%.
4. The actual accuracy achieved (Accuracy) is 0,9438.

Successful operation of the system requires compliance with the following conditions:

1. Preparation of the environment – complete installation of all software dependencies and provision of the necessary hardware power.
2. Quality of instructional data – the use of high-quality, consistent, and correctly labelled data, as well as the relative balance of classes to achieve the best results.
3. Deployment Procedure – Strict Sequence: Training → Verification → Classification of New Data.

The Sphere2 system is a specialised machine learning tool designed to binary classify users of social networks and messengers as "collaborators" or "non-collaborators". The classification is based on a comprehensive analysis of text messages, as well as the author's behavioural and temporal activity patterns. The goal is to integrate linguistic and metadata to build a robust user profile. The Data Processing and Preparation Module is responsible for collecting, consolidating, and initially cleaning the input data for further analysis (Table 5). A hybrid model integrating text and numerical features is used to classify the user (Table 6–7). The model training and validation module provides training and selection of the optimal classifier (Table 8).

Table 5
Data Processing and Preparation Module

Subfunction	Action	Description
Downloading data	Consolidation and validation of sources.	Support for .xlsx format. Automatic merging of the primary and additional datasets. Mandatory validation of the presence of key fields: 'marking', 'author id', 'message text'.
Text cleaning and normalisation	Noise removal and standardisation of text.	Includes removing URLs, mentions (@username), hashtags (#), normalising spaces, and lowercase text, and handling of Missing Values (NaN).
Dataset balancing	Preparation of data at the author level for training.	Merge messages into an author's profile. Automatic labelling of authors based on a configurable threshold. Duplicate deletion.

Table 6
Feature engineering module

Type of signs	Specification	Parameters
Textual features	Vectorisation of the consolidated body of the author's messages.	TF-IDF vectorisation at max. 10,000 signs. N-grams (1, 2) (unigrams and bigrams). Term filtering: minDF = 5, maxDF = 0.70.
Numerical features	Behavioural and metadata of activity.	Quantitative: total number of messages; average message length; number of unique groups/channels. Temporal: duration of activity (in days); activity intensity (messages/day).
Normalisation of signs	Standardisation of numerical characteristics.	Applying StandardScaler to numeric features. Saving scaler parameters.

Table 7
User retention and classification

Function	Description
Serialisation of components	Save the trained model, TF-IDF vectorizer, and StandardScaler in .pkl format to ensure reproducibility and fast deployment.
Batch classification	Handling multiple users. Grouping messages by 'author id', feature extraction, binary classification with configurable threshold (default 0.5).
Confidence score	Calculation of the confidence level of the forecast based on the deviation of the probability from the threshold of 0.5.

Table 8
Model Training and Validation Module

Function	Detail
Algorithm support	Multi-algorithmic approach support: Logistic Regression, Random Forest, SVM. Automatic selection of the best model according to the AUC (Area Under the Curve) metric.
Validation and evaluation	Using 5-fold cross-validation. Stratified division into training/test samples (80/20). Calculation of AUC-ROC, Classification Report, and Confusion Matrix metrics.
Class Imbalance Handling	Using a built-in mechanism $class_{weight}$ = balanced, in models to compensate for potential class imbalances.

The system provides tools for visual and statistical analysis of data and modelling results (see Table 9).

1. Visualisation: message distribution histograms, collaborative message fractions, class ratio diagrams, ROC-curves, and error matrices.
2. Statistical Analysis: Analysis of the balance of the dataset, calculation of the class imbalance coefficient, and statistics on data sources.

Table 9
Specification of input and output data

Data Type	Format	Required fields
Incoming (training)	Excel (.xlsx)	'marking' (0/1), 'author id', 'message text'
Input (classification)	pandas DataFrame	'Message Text'
Weekend (classification)	Dictionary	'is_collaborator' (bool), 'probability' (float), 'confidence' (string)

Technical and Performance Limitations:

1. The minimum requirements are a minimum of 1 message for user classification and a maximum number of TF-IDF traits of 10,000.
2. Language restrictions – support is optimised exclusively for the Ukrainian language.
3. Performance – recommended max. Number of authors for batch processing: 10,000. Classification time per user: ≤ 5 seconds.

To ensure the validity of the results, several conditions must be met, including the balance of classes and the validity of the results, as well as the class ratio in the study sample. The minimum number of samples for each class, ≤ 1:10, is 100. The system does not guarantee 100% accuracy. The results should be interpreted with consideration for the level of confidence. Regular expert and methodological verification is necessary. The criteria for the successful functioning of the model are the achievement of the following minimum metric values on the validation sample:

1. "AUC-ROC" ≥ 0.75.
2. "Precision" left (for the "collaborator" class right) ≥ 0.70.
3. "Recall" left (for the "collaborator" class right) ≥ 0.65.
4. F1-score≥ 0.67 (for both classes).

The leading actors for the first model (message classification, Fig. 1):

1. Data Scientist – develops and trains the model (the first script).
2. Data Analyst – uses the model to label large datasets (second script).
3. Developer – integrates the model into applications.
4. End User – uses to classify individual texts.

The two main processes for the first model (message classification, Figure 1):

1. Model development (training pipeline): Loading training data from 3 Excel files; Pre-processing and merging of datasets; Ensemble training (Naive Bayes and Logistic Regression); Evaluation and preservation of the model.
2. Using the model (Inference Pipeline): Loading the trained model; Batch classification of large datasets (400,000 records); Analysis of results and confidence statistics; Saving marked datasets.

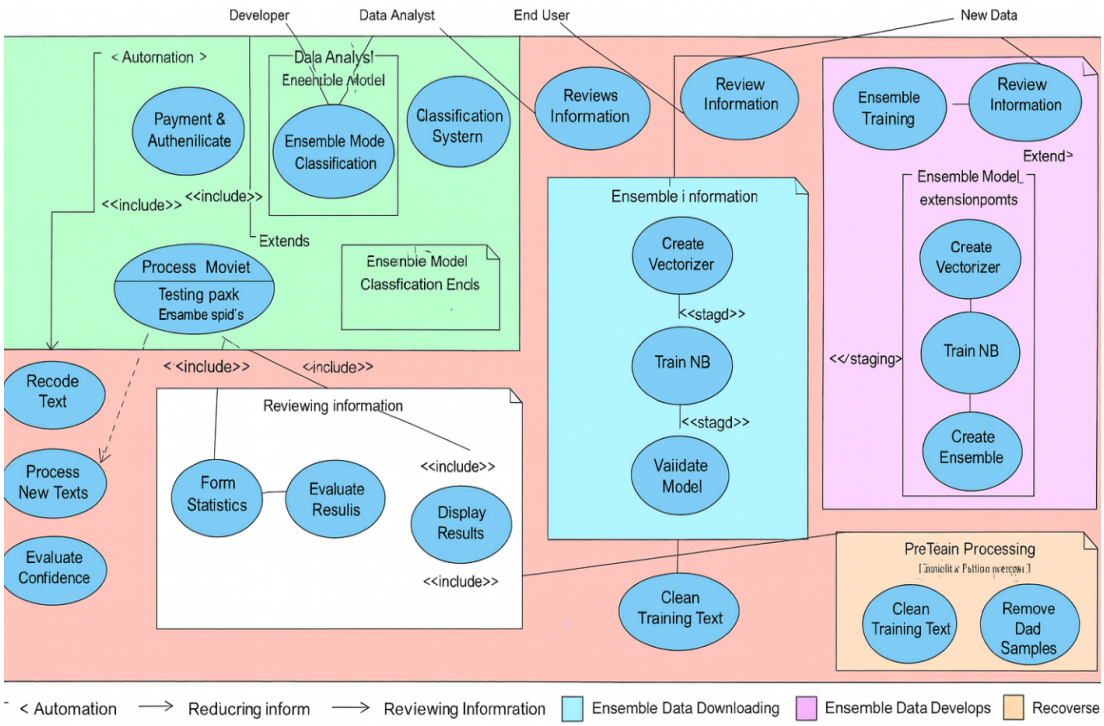

Figure 1: Use case diagram for the first model (message classification).

The main components for the first model (message classification, Fig. 2):

1. TextPreprocessor – responsible for preprocessing the text (lowercase casting, removing extra spaces).
2. DatasetLoader – loads and merges different datasets (ukr_messages_mark.xlsx, 2000_messages_labeled_complete.xlsx, messages_noemoji.xlsx).
3. FeatureExtractor – creates features using TfidfVectorizer and CountVectorizer.
4. EnsembleClassifier – contains two models (Naive Bayes with Logistic Regression) and performs ensemble prediction.

5. ModelTrainer – coordinates the process of training models.
6. ModelPersistence – Responsible for saving and loading models.
7. PredictionEngine – performs predictions for new data.
8. ResultAnalyzer – analyses the results of forecasting.
9. MainApplication is the main class that coordinates all processes.

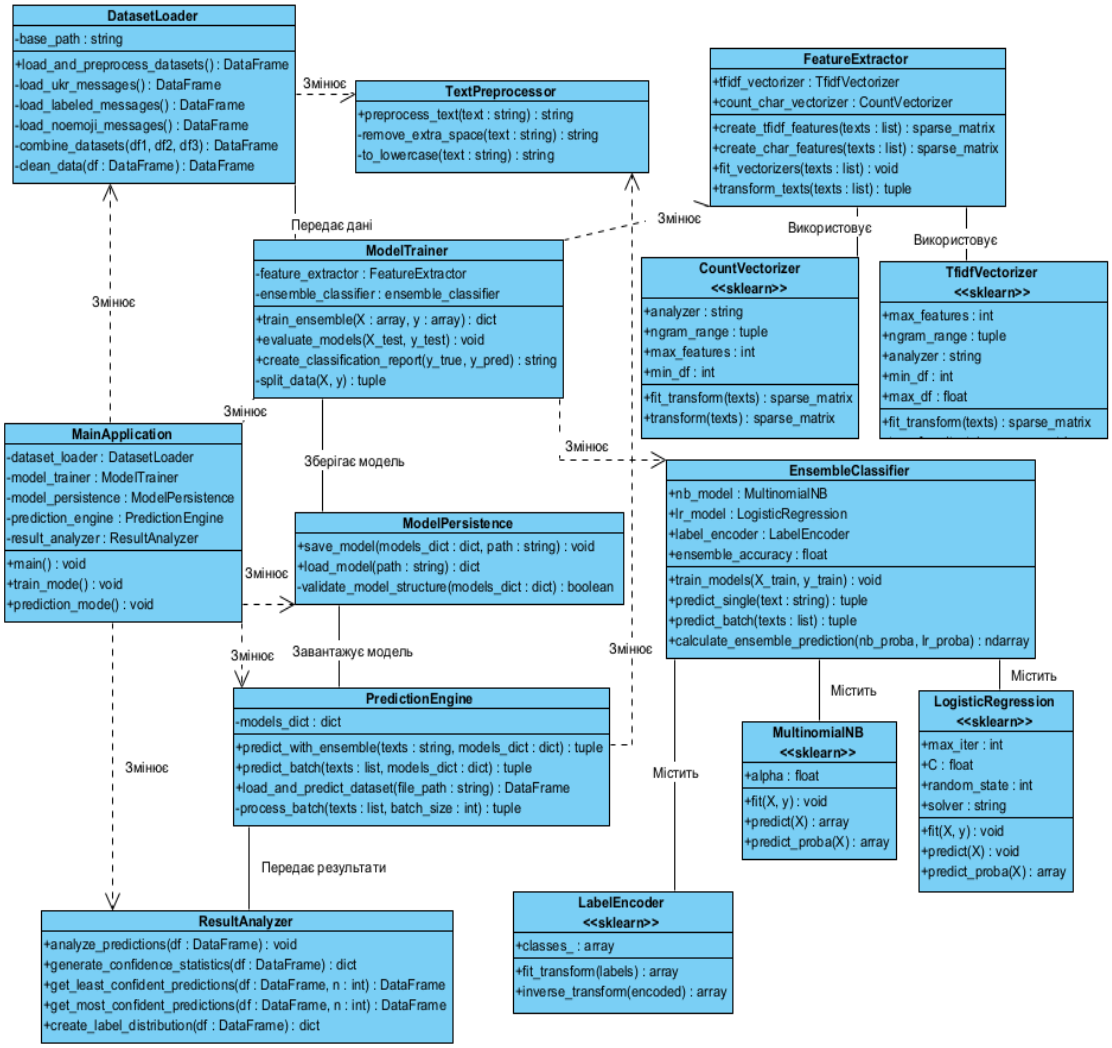

Figure 2: Class diagram for the first model (message classification).

Key features of the architecture for the first model (message classification):

1. The ensemble approach combines two different models with distinct vectorizers.
2. Modularity – each component has a clearly defined responsibility.
3. Batch processing – support for efficient processing of large amounts of data.
4. Persistence – the ability to save and load trained models.

Actors (system users) for the second model (account classification, Fig. 3):

1. The security analyst is the primary user for identifying potential collaborators.
2. Community moderator – a person who uses moderation tools to manage online communities.
3. Researcher – works with model training and evaluation.
4. System administrator – manages the technical aspects of the system.
5. An API client is an automated system that uses a model.

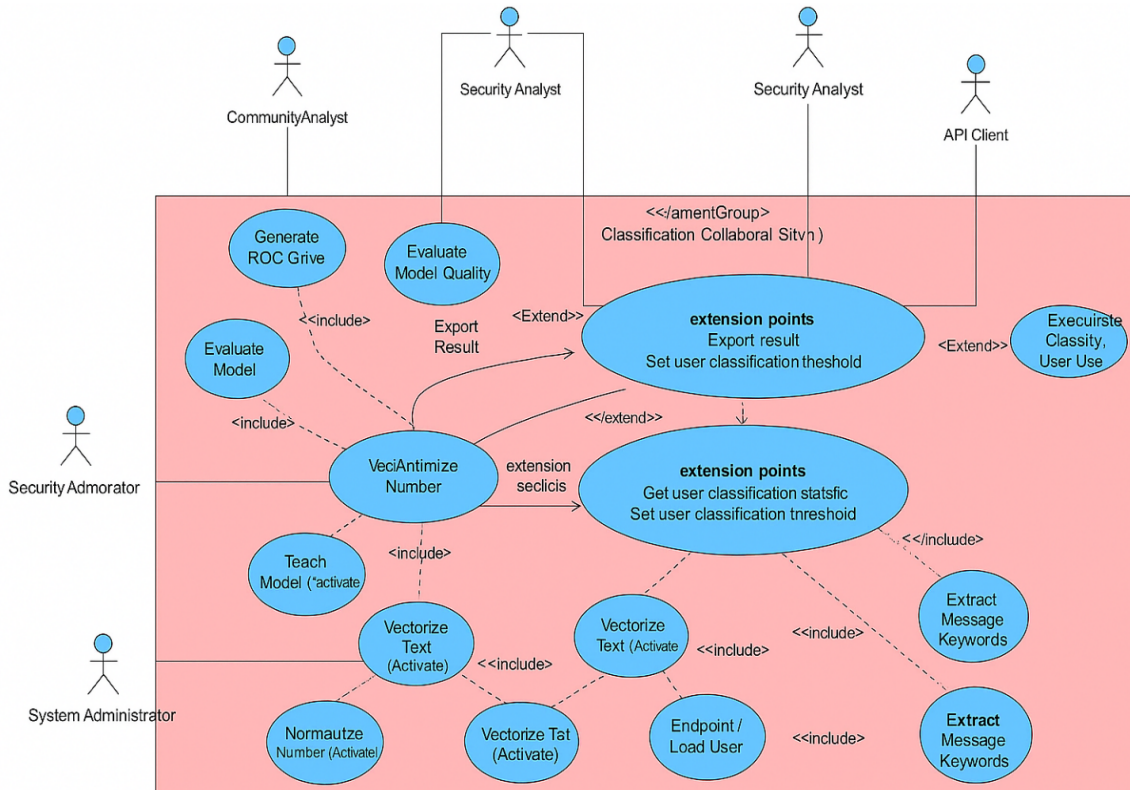

Figure 3: Use case diagram for the second model (account classification).

The main use cases for the second model (account classification):

1. Training and setup: Train the model on new data; Evaluate the quality of the model; Adjust the classification threshold; Save/load the model.
2. Classification: Classify an individual user; Batch classification of multiple users; Analyse user messages.
3. Analytics: Get classification statistics; Generate ROC curves and confusion matrices; Export results.
4. Auxiliary functions: Cleaning text data; Extracting features from messages; Text vectorization; Normalisation of numerical features.

The main classes of the class diagram for the second model (account classification, Fig. 4):

1. CollaboratorClassifier is the main class of the system responsible for classifying users as collaborators. Contains loaded ML models, TF-IDF vectorizer, and scaler for feature normalisation. Performs text cleaning, feature extraction from messages, and probability prediction.
2. BaseClassifier is an abstract base class that defines a standard interface for all machine learning models (fit, predict, predict_proba).
3. LogisticRegression, RandomForestClassifier, and SVC are specific machine learning models that inherit from the BaseClassifier class. Each model has its own particular parameters (random_state and class_weight) and implements both training and forecasting methods.
4. TfidfVectorizer – Responsible for converting text to numeric vector representations using the TF-IDF method. Configured to work with the Ukrainian language with parameters for n-grams and filtering rare/frequent terms.
5. Pipeline – creates sequential data processing pipelines that combine vectorisation and classification steps into a single process.

Utility classes for the second model (account classification):

1. ModelTrainer coordinates the process of training models, performs cross-validation, compares different algorithms, selects the best model, and stores it.
2. DataProcessor – is responsible for loading Excel files, cleaning text data (removing URLs, mentions, hashtags), splitting into training/test samples, and vectorising text.
3. ResultsManager – manages classification results, saving them in Excel format, generating classification reports, and building ROC curves and confusion matrices for visual analysis.
4. FileAnalyzer is a high-level class for analysing Excel files, which coordinates the entire process: data upload, individual analysis of each user, and the formation of final results.

Key features of the architecture for the second model (account classification):

1. Multi-model approach – the system supports three different ML algorithms (Logistic Regression, Random Forest, SVM) with automatic selection of the best.
2. Combined features are a combination of text features (TF-IDF) with numerical metrics (number of messages, average length, activity).
3. Modularity – a clear division of responsibility between components (data processing, training, analysis of results).
4. Persistence – saving trained models and vectorizers for later use.
5. Batch processing is an effective method for analysing data from multiple users in a single Excel file.
6. Reliability – error handling and test mode in the absence of trained models.
7. Multilingualism is an adaptation of the Ukrainian language with special rules for text cleaning.

Terms of reference for model 1 (classification of messages):

1. Accuracy ≥ 94%.
2. F1-score for each class ≥ 0.92.
3. Support for processing large amounts of data (100,000+ samples)
4. Ability to work with various data sources (.xlsx files).

Terms of reference for model 2 (classification of accounts):

1. AUC-ROC ≥ 0.97.
2. Precision for the "collaborator" class ≥ 0.82.
3. Recall for the class "collaborator" ≥ 0.97.

Architecture:

1. An ensemble approach, combining different algorithms.
2. Support for various text vectorisation methods.
3. Modular architecture for easy component upgrades.

Data processing:

1. Pre-processing of the Ukrainian text.
2. Removal of URLs, mentions, and extra characters.
3. Normalisation of case and spaces.
4. Vectorisation using TF-IDF and Count Vectorizer.

Algorithms:

1. Naive Bayes with TF-IDF vectorisation.

2. Logistic Regression with symbolic vectorisation.
3. Ensemble methods for combining results.
4. Support Vector Machine as an alternative algorithm.

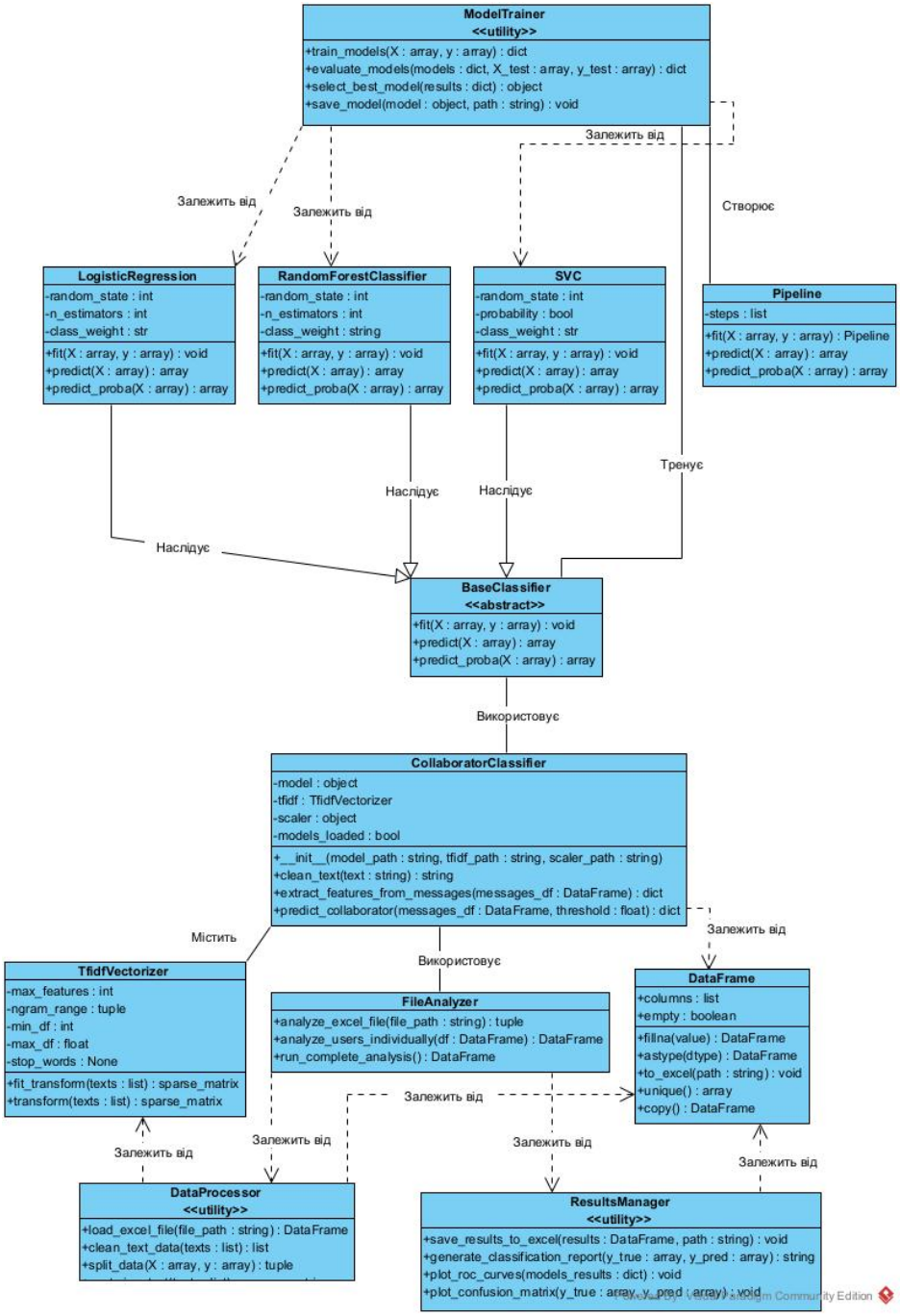

Figure 4: Class diagram for the second model (classification of accounts).

4. Selection of methods and means of the product being developed

This section presents a systematic analysis of experimental results obtained during training and a comparison of models for the binary classification problem of social media accounts. The purpose of this study is to determine the optimal model that achieves the highest accuracy in detecting collaborationist accounts under conditions of limited labelled data availability, significant class imbalance, and high text message noise levels. More than thirty combinations of vectorisation methods and machine learning algorithms are considered, including various options for adjusting sensitivity thresholds and class weighting factors.

The development of an information system for automatically detecting collaborationist messages and classifying accounts in social networks requires the use of methods capable of working with large arrays of short, noisy, and stylistically heterogeneous text data. As part of the study, a unique multi-source corpus was formed, comprising a manually marked dataset of 12,893 examples, as well as two automatically labelled datasets with a volume exceeding 140,000 examples. And 400 thousand messages, respectively. Such data properties determine special requirements for the choice of processing methods: support for multilingualism (Ukrainian and Russian), resistance to spelling variations, the ability to capture signs of transliteration, distortions, jargon, and short messages. With this in mind, the requirements for modelling have been formed:

1. High-quality classification on data of various natures.
2. Interpretation of results, necessary for analytical application.
3. Moderate computing costs.
4. Scalability to process hundreds of thousands of texts.
5. Resistance to class imbalances and to noisy data.

Taking into account these criteria, the study conducted a step-by-step comparative analysis of vectorisation methods and classification algorithms, while also considering the potential use of modern depth models.

In the educational data, the distribution of classes was uneven: class 0 (non-collaborators) accounted for approximately 63,6%, while class 1 (collaborators) comprised 36,4%. Such an imbalance is typical of the real online environment and requires the use of weighting strategies, threshold correction, and specialised analysis of F1 metrics for each class. Accuracy indicators in binary models under such conditions are insufficient to conclude; therefore, F1-macro (balanced average) and F1 for class 1 were chosen as the key metrics, as the primary goal of the system is to achieve a balanced average. Pre-processing included case normalisation, text cleaning of links, mentions, and redundant characters, and tokenisation. As a result of the analysis, it was determined that classical vectorisation methods – TF-IDF and symbolic n-grams – are best suited for the corpus, where a significant portion of the content is represented by short and spelling-erroneous messages (Table 10). TF-IDF (unigrams and bigrams) provided an effective capture of general topics, keywords, and phrases characteristic of a collaborationist, neutral, or pro-Ukrainian position. Instead, symbolic n-grams (2–4 characters) proved to be critical for identifying speech patterns characteristic of Russian-language propaganda and content with intentional phonetic distortions. They also made it possible to compensate for inaccuracies related to the use of surzhyk, jargon and transliteration. A significant advantage of the combination of lexical and symbolic vectorizers was the ability to cover both semantic and formal features of the text without losing essential information. During the experiments, several algorithms were tested, including Multinomial Naïve Bayes, Logistic Regression, SVM, Random Forest, and deep neural models (as part of the analogue analysis). The results demonstrated that:

1. Multinomial Naïve Bayes performed best in combination with TF-IDF, as it is well-suited for modelling the distribution of words in short messages.
2. Logistic Regression, in combination with symbolic n-grams, made it possible to most accurately separate stylistically different types of speech.
3. Although SVM achieved high performance on valid data, it required significantly more resources and exhibited poorer stability on significant cases.
4. Random Forest has proven to be less effective due to its weak ability to work with high-dimensional feature spaces.

Ensembling two models achieved the best results – Naïve Bayes (TF-IDF) and Logistic Regression (char-based), which enabled the combination of their strengths. The ensemble demonstrated the highest accuracy and stability in large enclosures, as well as the lowest rate of

misclassifications in critical classes, particularly in distinguishing between collaborationist and non-collaborationist messages.

Within the framework of the comparative analysis, modern approaches described in the literature were examined, including XLNet, BERT models, graph neural networks (GNN), contextual autoencoders (VAE), and multimodal architectures. Despite their high performance in a number of studies, these models have significant limitations for this task:

1. High computational complexity renders them impractical for widespread use in analysing data arrays with volumes of hundreds of thousands of records.
2. There is a need for long-term, additional training on specialised data to achieve optimal performance in low-resource languages (Ukrainian/Russian).
3. The complexity of interpretation is crucial for the application of the model in security and government settings.
4. A tendency to contextual "smoothness", due to which models can miss local stylistic markers important for detecting propaganda.

These factors determined the feasibility of using an ensemble of classical models that provide an optimal balance between accuracy, stability and practical suitability.

Given the class imbalance in the initial dataset, augmentation techniques were applied, including increasing the number of examples from the smaller class and using weighting factors during training. The initial distribution of labels is Class 0 – 10829 and Class 1 – 2064. The final distribution of labels after augmentation: 0 – 10829 and 1 – 6192. Additionally, a system of categorising model confidence levels (low, medium, high, and very high) was implemented, allowing for the use of a human-in-the-loop mechanism to enhance the accuracy of critical decisions. Checking the quality of the model involved not only analysing metrics (accuracy, F1, and ROC-AUC) but also a manual expert assessment of classification errors, which enabled the identification of hidden patterns and optimisation of the ensemble.

Table 10
Comparison of methods based on actual experiments

Method/Model	Dignity	Flaws	Need for resources
TF-IDF and Naïve Bayes	High speed; effectiveness on short texts; noise resistance; Simple interpretation	Sensitivity to words that are not in the dictionary; Less depth of context	Low
Char *n*-grams and Logistic Regression	Perfectly captures transliteration, jargon, spelling variations; high accuracy; resistance to distorted text.	High dimensionality of features; needs regularisation	Medium
SVM	High accuracy on clean data; Good work with difficult decision boundaries	High computational complexity, poor scalability, and long training	High
Random Forest	Works without normalisation requirements; Noise resistant	Does not work well in high-sized spaces; Low accuracy in word problems	Medium/ High

BERT / mBERT (embeddings)	Deep understanding of context; Best Metrics in Big Problems	The need for GPUs, long-term additional training, and complex interpretation can skip local markers	Very high
XLNet / SOTA Transformers	Maximum quality in large systems; good multilingualism	Extremely high learning difficulty; risk of overtraining; Resource consumption	Extremely high
GNN / VAE	Work with user relationships, detecting not only textual but also behavioural patterns.	The need for graph data, the complexity of architecture, and the unsuitability for a simple text pipeline	High

As a result of a comparative analysis of the methods, it was demonstrated that the ensemble of Naïve Bayes (TF-IDF) and Logistic Regression (char n-grams) offers the best balance of accuracy, speed, practical stability, and interpretability of results. Unlike deep neural networks, this approach does not require significant resources and demonstrates high efficiency on large text corpora, making it optimal for the task of automatically detecting collaborationist content.

A comprehensive test of several machine learning classification algorithms was conducted to assess their application in the binary detection of collaborationist activity (class 1: collaborator; class 0: non-collaborator). Three variants of TF-IDF vectorisation were used:

1. tfidf word balanced (based on words).
2. tfidf char balanced (based on symbolic n-grams).
3. tfidf mixed (a combination of both approaches).

The Support Vector Machine (SVM), Logistic Regression, Random Forest, and Gradient Boosting classifiers were tested, each in two configurations: with standard balancing (_balanced) and with increased positive class sensitivity (_sensitive, with a dynamic threshold of 0.35). The educational sample is characterised by a significant imbalance, where the share of the negative class (class 0) is approximately 63.6% (8663 samples), and the share of the positive class (class 1) is approximately 36.4% (4953 samples).

Let's describe the results of model training for binary classification for ten models (Table 11): (1) svm_balanced, (2) svm_balanced_sensitive, (3) svm_manual_weight, (4) svm_manual_weight_sensitive, (5) logistic_balanced, (6) logistic_balanced_sensitive, (7) random_forest_balanced, (8) random_forest_balanced_sensitive, (9) gradient_boosting, and (10) gradient_boosting_sensitive. The top 10 combinations by F1-macro are highlighted in bold. The best combination for the corrected score is the vectorizer tfidf_char_balanced – model svm_manual_weight (model 3, highlighted in the table in italics).

Table 11
Evaluation of models for binary classification

Model	1	2	3	4	5	6	7	8	9	10
Dynamic threshold for 1	0.400	0.350	0.400	0.350	0.400	0.350	0.400	0.350	0.400	0.350
Class Distribution in Learning Data{1.0: 4953, 0.0: 8663}										
Class percentages{np.float64(0.0): 0.6362367802585194, np.float64(1.0): 0.3637632197414806}										
Vectorizer tfidf_word_balanced										

Accuracy	0.9333	0.9319	0.9815	0.9809	0.8990	0.8822	0.9786	0.9721	0.9195	0.9222
F1-macro	0.9281	0.9269	0.9801	0.9795	0.8932	0.8773	0.9770	0.9702	0.9113	0.9152
F1-weighted	0.9334	0.9321	0.9815	0.9810	0.9000	0.8840	0.9786	0.9722	0.9187	0.9218
F1-binary(1)	0.9087	0.9079	0.9749	0.9741	0.8685	0.8528	0.9710	0.9627	0.8844	0.8908
F1 Class 0	0.9475	0.9459	0.9854	0.9849	0.9180	0.9018	0.9830	0.9777	0.9383	0.9395
F1 Class 1	0.9087	0.9079	0.9749	0.9741	0.8685	0.8528	0.9710	0.9627	0.8844	0.8908
Prediction*n* distribution {0:X, 1:Y}	{2159, 1246}	{2124, 1281}	{2139, 1266}	{2133, 1272}	{2028, 1377}	{1919, 1486}	{2123, 1282}	{2099, 1306}	{2274, 1131}	{2217, 1188}
Vectorizer tfidf_char_balanced										
Accuracy	0.9430	0.9421	0.9862	0.9844	0.9292	0.9210	0.9715	0.9615	0.9633	0.9612
F1-macro	0.9390	0.9382	0.9852	0.9833	0.9250	0.9170	0.9696	0.9592	0.9608	0.9587
F1-weighted	0.9432	0.9425	0.9862	0.9845	0.9298	0.9219	0.9717	0.9619	0.9635	0.9615
F1-binary(1)	0.9232	0.9227	0.9813	0.9790	0.9071	0.8987	0.9620	0.9494	0.9509	0.9484
F1 Class 0	0.9547	0.9538	0.9891	0.9876	0.9429	0.9353	0.9772	0.9689	0.9707	0.9689
F1 Class 1	0.9232	0.9227	0.9813	0.9790	0.9071	0.8987	0.9620	0.9494	0.9509	0.9484
Prediction*n* distribution {0:X, 1:Y}	{2118, 1287}	{2095, 1310}	{2131, 1274}	{2125, 1280}	{2051, 1354}	{1989, 1416}	{2093, 1312}	{2053, 1352}	{2099, 1306}	{2084, 1321}
Vectorizer: tfidf_mixed										
Accuracy	0.9307	0.9248	0.9812	0.9786	0.8966	0.8816	0.9747	0.9671	0.9204	0.9213
F1-macro	0.9251	0.9192	0.9798	0.9770	0.8904	0.8766	0.9730	0.9649	0.9124	0.9139
F1-weighted	0.9307	0.9250	0.9813	0.9786	0.8975	0.8834	0.9748	0.9673	0.9196	0.9208
F1-binary(1)	0.9046	0.8979	0.9745	0.9710	0.8644	0.8516	0.9660	0.9562	0.8860	0.8886
F1 Class 0	0.9456	0.9405	0.9851	0.9830	0.9165	0.9016	0.9799	0.9737	0.9389	0.9391
F1 Class 1	0.9046	0.8979	0.9745	0.9710	0.8644	0.8516	0.9660	0.9562	0.8860	0.8886
Prediction*n* distribution {0:X, 1:Y}	{2170, 1235}	{2136, 1269}	{2136, 1269}	{2125, 1280}	{2048, 1357}	{1929, 1476}	{2114, 1291}	{2088, 1317}	{2267, 1138}	{2238, 1167}

Word-based vectorisation (using unigrams and bigrams) yielded moderate results for most models. Combinations with SVM yielded F1-macro ≈ 0.93, while Logistic Regression yielded 0.88–0.90. The highest results were achieved by SVM variants with manually tuned weights, which reached an F1-macro score of approximately 0.9801. The addition of a sensitive threshold (0.35) had almost no effect on the F1 value, but increased the proportion of Class 1 predictions. This effect is expected: lowering the threshold increases recall for Class 1, but only slightly lowers the precision. Random Forest in combination with TF-IDF word-based showed high results (up to F1-macro 0.9770). However, models of this type are slower and more sensitive to a large number of features, which makes them less suitable for operational analysis of arrays of hundreds of thousands of texts. Gradient Boosting with verbal vectorisation yielded a significantly lower quality – the F1-macro score did not exceed 0.92. It is because boosting algorithms work less efficiently in the high-dimensional spaces formed by TF-IDF.

A significant improvement in quality was observed when transitioning to symbolic vectorisation (char n-grams), which enables the consideration of morphological distortions, transliteration, slang, propaganda, and toxic stylistic markers. This type of sign is critical for analysing the content created by enemy accounts. SVM in combination with TF-IDF char-based showed consistently high results:

1. Balanced F1-macro approximately 0.939.
2. Sensitive F1-macro approximately 0.938.
3. Manual weight F1-macro approximately 0.9852 (record).
4. Manual weight with sensitivity: F1-macro approximately 0.9833.

It was the combination of tfidf_char_balanced with svm_manual_weight that performed the best among all the models tested, yielding the following results: Accuracy 0.9862, F1-macro 0.9852, F1-class1 0.9813, and F1-class0 0.9891. It is significantly higher than the results of Logistic Regression and Random Forest with the same vectorizer. Logistic Regression in the char-based variant yielded an F1-macro in the range of 0.92–0.93, which is lower than SVM due to LR's weaker ability to model nonlinear boundaries between classes. Random Forest performed well (up to F1-macro 0.97) but failed to outperform SVM. The main reason is that the random forest does not scale well in high-dimensional TF-IDF spaces, where the number of features can reach thousands. Gradient Boosting in the character-based variant showed an F1-macro score of around 0.96, but fell short of the results achieved by SVM and Random Forest.

A vectorizer that combines verbal and symbolic features produced results between those of the previous two methods. SVM achieved an F1-macro score of approximately 0.93 for standard parameters and up to approximately 0.98 in manual weight mode. Random Forest scored 0.9730–0.9747, which is close to its char-based variant, but still inferior to SVM. Gradient Boosting again showed reduced efficiency (0.9124–0.9213 F1-macro). The best mixed combination (svm_manual_weight) achieved an F1-macro score of approximately 0.9798, which is a high score but 0.005–0.006 lower than the char-based SVM.

Among the 30+ models tested, the best were:

1. tfidf_char_balanced and svm_manual_weight – F1-macro 0.9852.
2. tfidf_char_balanced and svm_manual_weight_sensitive – F1-macro 0.9833.
3. tfidf_word_balanced and svm_manual_weight – F1-macro 0.9801.
4. tfidf_mixed and svm_manual_weight are F1-macro 0.9798.
5. tfidf_word_balanced and random_forest_balanced – 0.9770.
6. tfidf_mixed and random_forest_balanced – 0.9730.
7. tfidf_word_balanced and random_forest_balanced_sensitive – 0.9702.
8. tfidf_char_balanced and random_forest_balanced – 0.9696.

All the best results are related to SVM (manual weight), which confirms the high efficiency of this method in text spaces with a large number of features (Table 12).

Table 12
Comparison of models

Model	Principle of operation	Dignity	Flaws	Typical results
SVM (balanced)	Linear classifier with automatic class weights	Stability, high quality, and it works well in large spaces	Less accurate without manual weighing	F1 ≈ 0.93
SVM (balanced_sensitive)	Same + Lowered Threshold	Greater Class 1 recall	Slightly less precision	F1 ≈ 0.93
SVM (manual_weight)	SVM with hand-picked scales	Highest accuracy of all methods	Accurate selection of weights is required	F1 ≈ 0.9852
SVM (manual_weight_sensitive)	Same but lower threshold	More Class 1 Detections	Easy drop-in accuracy	F1 ≈ 0.983
Logistic Regression (balanced)	Linear model with scales	Quick, easy	Weak in nonlinear problems	F1 ≈ 0.89
Logistic Regression (sensitive)	Lowered threshold	More positive ones	Lower precision	F1 ≈ 0.88
Random Forest	Combination of trees	Strong classical method	Worse in high-sized spaces	F1 ≈ 0.97
Random Forest Sensitive	Lower threshold	Best recall	Possible FPs	F1 ≈ 0.96–0.97
Gradient Boosting	Successive trees	Good on tabular data	Weak in sparse TF-IDFs	F1 ≈ 0.91–0.92
Gradient Boosting Sensitive	Sensitive threshold	Increase recall	Does not improve overall quality	F1 ≈ 0.91

Symbolic TF-IDF (char-level) is the most efficient type of vectorisation for the task of detecting collaborationist accounts. SVM with hand-picked class weights demonstrates the highest stability and accuracy. Random Forest provides good results, but cannot outperform SVM in high-dimensional spaces. Logistic Regression loses to SVM due to the limitations of the linear model. Gradient Boosting is impractical for sparse high-dimensional text features. Lowering the classification threshold (sensitive) increases the recall of class 1, but does not give a significant improvement in F1-macro. The best model, after selection, was further trained on the entire dataset to maximise its generalising ability. The analysis of experimental results demonstrated that the

optimal combination for the task of automatically detecting collaborationist accounts is the SVM model with manual scales, in combination with TF-IDF symbolic vectorisation. This approach provides the highest-quality metrics, scales perfectly, works quickly on large amounts of data, and remains interpretable, making it the most suitable for practical application in information security analysis systems. For the comparative analysis, the weighted F1-macro metric was used, which is sensitive to performance on both classes (Table 13). Based on the obtained metrics, the following combination demonstrates the best performance: a vectorizer (symbolic n-grams) and a model (method of reference vectors with manually weighted factors). This configuration achieved the highest scores: an overall accuracy of 0.9862, a weighted F1 measure (F1-macro) of 0.9852, and an F1 measure for the positive class (Class 1 – collaborators) of 0.9813. The high Accuracy and F1-macro scores, as well as the virtually identical F1 values for both classes (Class 0: 0.9891, Class 1: 0.9813), indicate the exceptional generalisation capacity of the model and the practical overcoming of the problem of class imbalance.

Table 13
Comparison of models

Vectorizer	Model	Acc.	F1-macro	F1 (Class 1)	Dynamic Threshold	Distributio *n* of forecasts (1/total)
tfidf_char_ balanced	svm_manual_ weight	0.9862	0.9852	0.9813	0.400	1274/3405
tfidf_char_ balanced	svm_manual_ weight_sensitive	0.9844	0.9833	0.9790	0.350	1280/3405
tfidf_word_ balanced	svm_manual_ weight	0.9815	0.9801	0.9749	0.400	1266/3405
tfidf_mixed	svm_manual_ weight	0.9812	0.9798	0.9745	0.400	1269/3405
tfidf_word_ balanced	svm_manual_ weight_sensitive	0.9809	0.9795	0.9741	0.350	1272/3405
tfidf_word_ balanced	random_forest_ balanced	0.9786	0.9770	0.9710	0.400	1282/3405
tfidf_char_ balanced	random_forest_ balanced	0.9715	0.9696	0.9620	0.400	1312/3405
tfidf_char_ balanced	Gradient_ boosting	0.9633	0.9608	0.9509	0.400	1306/3405
tfidf_word_ balanced	svm_balanced	0.9333	0.9281	0.9087	0.400	1246/3405
tfidf_char_ balanced	logistic_balanced	0.9292	0.9250	0.9071	0.400	1354/3405

There is a clear advantage of vectorisation based on symbolic *n*-grams (tfidf char balanced) over word-based vectorisation (tfidf word balance) and mixed approach (dtfidf mixed). For example, the best model svm manual weight on tfidf char balance (F1-macro = 0.9852) outperforms the same

model on dtfidf word balanced (F_1-macro = 0.9801). It supports the hypothesis that using symbolic n-grams is vital for accounting for slang, transliteration, and spelling variability in informal textual data. The SVM (Manual Weight) classifier, optimised with manual or weighted parameters, yields the best results in all vectorisation configurations, indicating its high efficiency in the TF-IDF feature space. Although Random Forest showed high accuracy (up to 0.9786), it is inferior to optimised SVM models. Gradient Boosting (up to 0.9222) and Logistic Regression (up to 0.8990) performed lower. Lowering the dynamic threshold to 0.35 (sensitive models) generally resulted in a slight decrease in F1-macro. However, these models are important for scenarios where the priority is to maximise the completeness (Recall) of the positive class at the expense of a slight increase in False Positives. It has been established that for the task of binary classification of users as potential collaborators, the optimal combination of TF-IDF vectorisation based on symbolic n-grams and the Support Vector Machine classifier with optimised weighting factors is the most effective approach. The achieved Accuracy of 0.9862 and F1-macro 0.9852 confirm the high reliability of the developed approach.

After identifying the best model at the validation stage (based on the F1-macro and Accuracy metrics), a predictive classification was carried out on an extensive array of unlabeled data.

1. Distribution of predictions: Class 0 (Non-Collaborator) 63.383 samples (71.3%) and Class 1 (Collaborator) 25.481 samples (28.7%).
2. Medium Confidence: Class 0 – 0.965 and Class 1 – 0.880.

A high average level of confidence for both classes indicated potentially high reliability of the model. However, absolute performance metrics on a test sample do not always correlate with satisfying specific classification requirements in sensitive domains.

Despite the high quantitative metrics, further qualitative verification of labelling revealed a significant issue related to the system's ethical and functional requirements: incorrect classification of pro-Ukrainian content. It turned out that the leader model, optimised for the F1 metric, is prone to mistakenly classify pro-Ukrainian messages explicitly as Class 1 (Collaborationist). It is a critical False\ Positive error that contradicts the main goals of the project and carries high risks during operation. It is because pro-Ukrainian content, like collaborationist content, can contain a high proportion of aggressive language or specific terms, which confuses linear models that use TF-IDF.

Due to the identified discrepancy, the model selection process was changed: the priority was not to maximise the overall F1 measure, but to minimise False\ Positive errors for pro-Ukrainian content. Iterative testing of various models for quality markings was carried out. The naive_bayes_strict model showed the best results in terms of distinction: pro-Ukrainian messages were correctly classified as Class 0 (Neutral/Non-Collaborative). A new problem arose: a significant number of neutral messages began to be mistakenly classified as Class 1 (Collaborationist). This result highlighted the need to strike a balance between sensitivity to pro-Ukrainian content and the accuracy of categorising neutral and collaborationist messages.

To achieve the necessary balance, it was decided to use the Ensemble of Classifiers. The ensemble approach allows you to combine the advantages of different models:

$$P_{\text{Ensemble}}(C=1)=\text{Average}\left(P_{\text{NB}},P_{\text{LR}}\right),$$

where P_{NB} is the probability from Naive Bayes, and P_{LR} is from Logistic Regression.

The optimal ensemble consists of Naive Bayes (with TfidfVectorizer) and Logistic Regression (with CountVectorizer, character-level). The Naive Bayes classifier (with TfidfVectorizer) is effective for working with lexical features and demonstrating high resistance to the erroneous classification of pro-Ukrainian content as a positive class. The Logistic Regression classifier (with CountVectorizer, char-level) adds linear discriminant power by using symbolic n-grams to increase sensitivity to stylistic and spelling patterns (which is vital for identifying the specific vocabulary of collaborators). This combination ensured the simultaneous achievement of three critical criteria:

1. Correct classification of collaborationist messages.
2. Minimising the erroneous classification of pro-Ukrainian content as collaborationist.

3. Reducing the level of misclassification of neutral messages as collaborationist.

For further optimisation, several meta-models were tested (Stacking Ensemble), where the predictions of the base classifiers were used as features for the final classifier (e.g. Naive Bayes, Logistic Regression or Gradient Boosting). For further labelling and operation, the Naive Bayes Ensemble (TfidfVectorizer) and Logistic Regression (CountVectorizer, char-level) were selected as the most balanced solution that meets both quantitative and critical qualitative classification requirements in the sensitive domain.

5. Software development and description of its structure

The developed software is designed for the automated processing of large amounts of text data from social networks and the further classification of messages and accounts based on the level of activity of collaborationists. Since the analysis is carried out on real Telegram cases, which can contain hundreds of thousands of records, the key requirements for the system are scalability, modularity, resistance to noisy data, model interpretation, and the possibility of batch processing. The software architecture is modular and consists of several logically isolated subsystems:

1. Text preprocessing module.
2. Data vectorisation modules.
3. Modules for training and saving models.
4. Ensemble classifier.
5. Batch forecasting module.
6. Subsystem of analysis and generalisation of results.

Each component is designed to be autonomously tested, replaced, or supplemented without interfering with other parts of the system. It ensures scalability and the ability to further expand the system. At the conceptual level, the system implements the classic machine pipeline (Table 14): *data loading → purification → vectorisation → model training → ensemble → forecasting → analysis of results.*

Table 14
Comparison of system components

Component	Dignity	Flaws	Limitations	Recommendations
TextPreprocessor	Unification of data structures; noise cleaning; Standardisation	Does not work with multimodality	Sensitivity to rare symbols	Add morphological analysis, stemming
TF-IDF Vectorizer	Interpretation; speed; Stability	Miss the context; it does not work with transliteration	Needs large dictionaries	Use subword models
Char *n*-grams	Resistance to distortion, bot style, and transliteration	Less semantics	High dimensionality of space	Optimise *n*-grams, add hashing
Naive Bayes	Fast, interpreted; great for short texts	Linearity of decisions; weak on complex semantics	Poor work with trait correlations	Use anti-aliasing and class weights
Logistic	More flexible	Sensitive to class	Needs	Use balanced class

Regression	boundaries; Stable results	imbalances	normalization	weights
Ensemble classifier	Increases accuracy; unites the strengths of the models	Complicates interpretation	Dependence on basic models	Add a weight ensemble
Batch Prediction Engine	Scalability; stability; fault-tolerant	Requires more memory	Latency on large files	Add Asynchrony
Result Analyzer	In-depth analytics; confidence-groups; Examples	Does not give a legal assessment	Varies by model	Integrate explainability (SHAP)
Model Persistence	Reproducibility, portability	Dependency on library versions	Requires versioning	Add MLflow or DVC

The Word Preprocessing Module is responsible for standardising data, including:

1. Removal of HTML artefacts, emojis, character repetitions.
2. Normalisation of the register.
3. Cleaning URLs, mentions, hashtags.
4. Combining different column formats into standard text and labels.
5. Extraction of empty strings and duplicate entries.

Since the corpora are collected from various sources (Telegram groups, moderator tables, and manually labelled samples), each input file may have a different column format. Therefore, the module automatically maps the names to standard ones, ensuring the uniformity of the input set.

Initially, a model was created to mark 8,864 messages, based on 2,000 manually marked messages from a collaborationist group, 2,000 messages from a pro-Ukrainian group, and 10,861 manually marked messages from one collaborator. In total, three datasets with 9,0896, 10,861, and 2,000 records were collected, respectively. Total number of records after merging: 14835. Label distribution: {'0.0':10762, '1.0':2061, '-1.0':2012}. The training sample consists of 11,868 records, and the test sample comprises 2,967 records. Evaluation of the Initiated Models:

1. Naive Bayes (Weighted Count) Accuracy: 0.8891.
2. Logistic Regression (Count Char) Accuracy: 0.9269.
3. Ensemble Accuracy: 0.9178.

To build a model in the feature vectorisation and extraction module, two types of features are used:

1. Verbal (TF-IDF, word n-grams) – to display semantics.
2. Symbolic (char n-grams) – for working with transliteration, distortion, jargon, and false spaces.

This combination allows the model to both "understand the content" and "understand the form". Vectorizers are trained only on the training sample and are stored in the model file along with the statistical parameters.

The developed product uses two different models in the basic model training module. Multinomial Naive Bayes is optimal for TF-IDF representations of short texts, as it is both fast and

interpretable. Logistic Regression – it better models symbolic features and provides more flexible boundaries for decision-making. For each model, the following is carried out:

1. Learning about the appropriate vectorization.
2. Evaluation on a test sample.
3. And saving metrics (Confusion matrix, Precision, Recall, F1).

It provides transparency and the ability to audit each model separately.

The ensemble of classifiers combines the probabilistic outputs of both models by averaging their results. Advantages of ensemble:

1. Reduction of model errors due to different types of features.
2. More stable forecasts on non-standard texts.
3. Increased interpretation (you can see which model gave the weaker signal).

The ensemble also generates a confidence score (confidence level) and models agreement (the degree of agreement between the models themselves). These values are used in the analysis module and in the definition of risk forecasts.

Given that the volume of real data can exceed hundreds of thousands of rows, forecasting is carried out in batches (batch inference) in the batch forecasting module. Module functionality:

1. Automatic determination of the optimal package size.
2. Error handling inside packages.
3. Return of standard values for incorrect records.
4. Progress bar and logging.
5. Saving the results to Excel or a DataFrame.

Batch ownership increases reliability and allows you to restore the process after failures.

The module for analysis and generalisation of results generates analytical reports, including:

1. Distribution of forecasts between classes.
2. Analysis of confidence levels by groups: low/mid/high/very high.
3. Calculation of average and median confidence.
4. Examples of the most and least reliable forecasts.
5. Assessment of model consistency.

This module plays a crucial role in the practical application of the results, as it enables manual analysis experts to identify which data requires additional revision.

The logic of the system and integration mechanisms, in particular, the Training pipeline:

1. Loading enclosures of various structures.
2. Normalisation of columns, cleaning, and filtering by text length.
3. Division into training and test samples with stratification.
4. TF-IDF vectorizer training → Naive Bayes training.
5. Char-n-grams training → Logistic Regression training.
6. Generation of metrics; selection of optimal parameters.
7. Building an ensemble.
8. Saving the model, vectorizers and encoders.

Inference pipeline:

1. Reading the input Excel file.
2. Validation of the table structure.

3. Text pre-processing.
4. Batch separation.
5. Getting predictions from both models.
6. Averaging probabilities.
7. Calculation of confidence and model agreement.
8. Write the results to a file.
9. Generation of a report and statistics.

Let's outline the primary technical limitations and associated risks. The model depends on the quality of Telegram cases. There may be a "drift" of content, which reduces efficiency over time. The system does not take into account multimodal data (images, videos). The model does not moderate or determine the legal assessment of the collaboration – only the text classification. Recommendations:

1. Regular retraining every 3-6 months.
2. Introduction of the human-in-the-loop method.
3. Expansion of the case on TikTok, Instagram, and YouTube comments.
4. LLM integration for semantic message reformulation.

6. Iterative development and validation of models for the classification of textual content and identification of authors in social networks

At the first stage, an ensemble classifier was developed for automatic labelling of a large data array (88,864 messages), which was critically necessary for further training of the model at the author level. The study sample is compiled from three sources: 2,000 messages from a collaborationist group, 2,000 messages from a pro-Ukrainian group, and 10,861 labelled messages from a single collaborator. The total amount of data after cleaning and merging was 14,835 unique records. The classification is multi-class (-1.0: pro-Ukrainian, 0.0: neutral, 1.0: collaborationist). Ensemble (Soft Voting):

1. Model 1 – Multinomial Naive Bayes (MNB) with TfidfVectorizer (based on words and bigrams).
2. Model 2 – Logistic Regression (LR) with CountVectorizer (based on symbolic *n*-grams).

An estimate on the test sample (2,967 samples) confirmed the effectiveness of the hybrid approach (Table 15). Despite the high overall accuracy (≈ 92%), the relatively low Recall for Class 1.0 (collaborationist) at 0.65, and the relatively low F1-score (0.76) indicate the model's moderate ability to detect a positive class.

Table 15
Ensemble evaluation results

Model	Accuracy	F1-macro	F1 (Class -1.0)	F1 (Class 1.0)
Naive Bayes (Word)	0.8891	-	-	-
Logistic Regression (Char)	0.9269	-	-	-
Ensemble (Final)	0.9178	0.86	0.88	0.76

At the second stage, the developed Ensemble was used for automatic labelling of a large array of data from 88,864 previously unlabeled messages. Distribution of predicted labels: 0.0 (neutral) – 68.4%; 1.0 (collaborationist): 30.5%; -1.0 (pro-Ukrainian) – 1.1%. Confidence Stats: Average Trust – 0.834, and in High Trust (>0.8): 66.8% of records. Accordingly, 82.9% of cases had full agreement (All_Agree) between the MNB, LR and the Ensemble, confirming the consistency of the classifiers. Examples with the lowest confidence (e.g., 0.345) and prediction discrepancy (Partial_Agree, Disagree) were identified as samples on the verge of decisions that require potential manual verification, e.g., the text: 'pshöl nahuy pös abossannyi...' Prediction: -1.0 (Confidence: 0.345, NB:-1.0, LR: 1.0). This indicates the difficulty of classifying aggressive but ambiguous vocabulary.

The third stage aimed to train the advanced message classification ensemble on a combined, qualitatively and auto-labelled dataset to ensure maximum accuracy before proceeding to author classification. The total volume of the auto-labelled dataset is 141,357 entries (including 88,864 auto-labelled ones). The label distribution is as follows: 0.0 – 50.6%, -1.0 – 28.7%, and 1.0 – 20.7%, respectively. On the test sample (28,272 samples), the new Ensemble (Naive Bayes with TfidfVectorizer and Logistic Regression with CountVectorizer) demonstrated a significant improvement in metrics, especially for less represented classes (Table 16). Through the expansion of the data corpus and the improvement of the ensemble, the model achieved an F1-score of 0.92 for the critically important class 1.0 (Collaborative), while maintaining a high Precision of 0.95, which minimises false positives. In the fourth phase of the study, the final message classification model was applied to label an array of 372,940 messages, forming the basis for the next author-level model. Distribution of forecasts respectively for 0.0 – 93.4%; 1.0 – 5.6%; -1.0 – 1.0%. Confidence Distribution – 58.2% of predictions have very high confidence (>0.9), and the average confidence is 0.8672. Low confidence – only 0.8% of records are classified with low confidence (0.0–0.5), confirming the model's reliability.

Table 16
Ensemble evaluation results

Class (Label)	Precision	Recall	F1–score	Support
-1.0 (pro-Ukrainian)	0.98	0.93	0.96	8116
0.0 (Neutral)	0.92	0.98	0.95	14317
1.0 (Collaborative)	0.95	0.88	0.92	5839
Accuracy			0.9438	28272
F1-macro			0.94	

Table 17
Results of the evaluation of the authors' classification model (assessment on a test of 366 samples)

Model	Test AUC	Accuracy	F1 (Class 1)	Precision (1)	Recall (Class 1)
SVM	0.9770	0.92	0.90	0.83	0.99
Random Forest	0.9769	0.91	0.90	0.83	0.99
Logistic Regression	0.9769	0.91	0.89	0.82	0.97

At the fifth stage, based on the labelled corpus, an educational sample was formed for author classification, which combined 372,940 bulleted messages with an additional body. An author is classified as a collaborator (Class 1) if the proportion of his collaborationist messages in the total activity (Collab Ratio) is 0,006 (empirically determined optimal threshold). Authors from the supplementary corpus are considered Class 0 (non-collaborators). Distribution of authors (1,827 in total): collaborators (Class 1) – 717, non-collaborators (Class 0) – 1,110. The ratio (balance) of Class 1 to Class 0 is 0.646, which is a significant improvement in balance over the original sample of messages. Comparative training of models at the author level was conducted using TF-IDF traits (aggregated text of the author, Table 17). Vectorisation is based on TF-IDF on words and bigrams (10,000 characters). SVM is chosen as the best model (AUC = 0.9770). The model exhibits a very high Recall (0.99) for detecting collaborators, although this results in a moderate decrease in Precision (0.83). To assess the impact of metadata, a combined model (Logistic Regression on TF-IDF and numerical characteristics, including the number of messages, average length, and activity intensity) was tested. The combo model exhibited a slightly lower AUC (0.9697) compared to the pure SVM on text features (0.9770). It indicates that linguistic features (aggregated text) are the most significant predictor of collaborationist activity.

7. User Classification Model Validation: Benchmark Examples and Performance Analysis

An analysis of the classifier's quality at the message level was conducted at the previous stages of iterative development. Here is an evaluation and analysis of control examples of CollaboratorClassifier, a model designed to binary classify user accounts as "Collaborator" or "Non-Collaborator". The model used for validation (collaborator_classifier_combined.pkl) is a combined classifier (Logistic Regression) that operates on both textual features (TF-IDF from the author's aggregated text) and numerical features (metadata normalised using StandardScaler). Input data for author classification (aggregated):

1. Text features are the combined and cleaned text of all user messages.
2. Numerical features – the total number of messages, the average length of the message, the number of unique groups, the duration of activity (in days), and the intensity of activity (messages per day).

To verify the minimisation of False Positive errors, a user profile with typical neutral activity was simulated (Tables 18–19).

Table 18
Input (neutrality scenario)

Field	Meaning	Description
Message Text	Hello everyone!, How are you?, Good weather today	Typical neutral/social content.
Date	2024-01-01, 2024-01-03	Activity for 3 days.
Group name	Group 1, Group 1, Group 2	Activity in 2 unique groups.
Extracted features (extrapolation):	Total Msgs: 3; Avg Msg Length: moderate; Unique Groups: 2; Activity Span: 3 days.	

The model correctly classified the user as a non-collaborator (Probability = 0.123, which is well below the threshold value of 0.5). The classification confidence is defined as High (|0.123 − 0.5| = 0.377 > 0.3). This result confirms that the model effectively identifies typical neutral activity and, due to the weighting of linguistic features, is not prone to erroneously classifying neutral profiles as positive of the predicted probability from the binary threshold of 0.5:

$$\text{Confidence} = \begin{cases} High, if\ |P-0.5| > 0.3, \\ Medium, if\ 0.15 < |P-0.5| \le 0.3, \\ Low, if\ |P-0.5| \le 0.15. \end{cases}$$

Table 19
Classification results

Metric	Meaning
Is_Collaborator	No (False)
Probability (Class 1)	0.123
Confidence	High
Message	The probability of a collaborator is 0.123

In the control example, the probability of 0.123 is at a considerable distance from the 0.5 threshold (in favour of Class 0), which logically leads to high confidence. Since the control on neutral data was successful, the following validation should be aimed at verifying:

1. Will the model be able to correctly classify the example of a collaborator with a minimum number of messages?
2. How will the model behave in the absence of some metadata (for example, data on the date of activity)?

Preliminary validation confirmed the model's reliability using a neutral example. At this stage, the focus is on testing the Cross-Domain Robustness of the model. Although the model was trained on Telegram data, its performance is evaluated on samples from Twitter, another social media platform. It is critical because changes in format and language patterns between domains often lead to degradation of classifier quality. Here and in the following examples, I will check whether the model correctly classifies users' accounts from Twitter, although the model learned from users' messages from Telegram, since in many studies related to this topic, models are poorly classified when they receive data from social networks other than those on which they were trained. Let's first check the accounts of two Twitter users: one with a pro-Russian stance and the other with a pro-Ukrainian stance, on an ensemble of models (Table 20). A combined model for author classification is used, which operates on aggregated text and metadata.

Table 20
Analysis of accounts of different authors

Eoristor	@mezyukho	@Valerii_Markus
Examples of messages	https://t.co/iXy3mP5f9U, https://t.co/cJEz3CKa8z, https://t.co/m7rF0b8moy, https://t.co/yF8z8Fhbew	https://t.co/ufJ2kU47Yl, https://t.co/ZrhRwyMcXK, https://t.co/KIdAThmt51

Quantity	50	50
Collaborator	Yes	Yes
Probability	0.937	0.058
Confidence	high	high
Medium length	236,9 characters	150,6 characters
Incoming Content	The messages contain direct support for aggression (#КНДР, special military operation, EU military assistance to Ukraine). Specific terminology is used, which is a strong feature of Class 1.	The messages contain patriotic language, calls for meetings (e.g., "Glory to Ukraine!", "Markus Foundation", drones). The presence of strong signs of the Pro-Ukrainian/Neutral class.
Conclusion	The model correctly classified the account of a collaborator, a traitor to Ukraine, as a collaborationist and did so with high confidence.	The model correctly classified Valery Marcus' account as non-collaborationist, and also with high confidence.

The model successfully classified the @mezyukho account as collaborative with high confidence (0,937). It confirms the effectiveness of TF-IDF and aggregated text in detecting content signs of a positive class, even when the domain is changed from Telegram to Twitter. The account @Valerii_Markus was correctly classified as non-collaborationist with high confidence (0,058). It suggests that the model can reliably distinguish between patriotic and collaborationist narratives, thereby avoiding the critical False Positive errors that were previously a problem in the message classification phase.

Let's conduct another analysis – specifically, examining the extreme cases (Table 21). Two contrasting profiles were used to evaluate the model's discriminatory ability.

Table 21
Analysis of accounts of different authors

User	@elonmusk	@pasha4mmm
Examples of messages	https://t.co/9hjQfNNAfC, https://t.co/xVjXBJ97nU, https://t.co/Z6kf6sj2mS	https://t.co/KEUxJtimY7, https://t.co/I5fwP4lfgx, https://t.co/3w4SXTWFkS
Quantity	49	50
Collaborator	Yes	Yes
Probability	0.211	0.779
Confidence	medium	medium
Medium	64.3 characters	89.1 characters

length		
Incoming Content	English-language content (technology, US politics). Language outside the learning sample.	Mostly neutral content (In the village for now, and soon in Moscow..., Good village morning). The possibility of "masking" through social topics that do not contain direct political signs.
Conclusion	The model, of course, correctly identified the account as non-collaborationist. Still, you should not expect high confidence from it, as it happened, the model has average confidence about this user.	The model correctly classified this user's account as collaborative. Medium confidence due to the small sample (50 messages) and due to the fact that this user writes a lot of messages about nothing.

The model correctly classified two collaborators as collaborators and one patriot as a non-collaborator with high confidence. Additionally, the model correctly classified even an English-speaking user as a non-collaborator, despite not having been trained on English-language data. As expected, the model classified the account as non-collaborationist, as it does not contain any linguistic features related to the Ukrainian-Russian conflict. However, confidence has dropped to "Medium" ($0.15 < |0.211 - 0.5| = 0.289$ 0.3). It confirms the model's sensitivity to unknown language spaces. Although the classification is correct, the absence of relevant features in the TF-IDF space prevents the highest level of confidence from being achieved. The model also correctly classified the account as collaborative (Probability = 0.779). The decrease in confidence to "Medium" (0.779 is closer to the threshold of 0.5 than 0.937) is due to the low informative content of the content (random household messages), which leads to weaker activation of the most significant signs in the TF-IDF space. It demonstrates that the quality of the content and its political focus directly affect the confidence level of the classifier. The cross-domain validation carried out on Twitter control examples confirmed the high stability and discriminatory ability of the CollaboratorClassifier model, including its cross-domain efficiency, sensitivity to traits, and linguistic limitations. The model trained on Telegram functions successfully on Twitter data, correctly classifying both opposing classes (collaborator and patriot) with high confidence. The high confidence of the forecast directly correlates with the presence of strong, politically colored features in the text. The model predicts an expected decrease in confidence in language processing that is not observed in the training sample (English), confirming the need for specialisation of NLP models.

8. Comprehensive analysis of the results of message classification and author identification

The presented analysis covers the evolution and final evaluation of two key models: the multiclass message classifier and the binary classifier of user accounts.

The purpose of the first model (Fig. 5 and Table 22) is to classify text messages into three categories: pro-Ukrainian (-1.0), neutral (0.0), and collaborationist (1.0). The F1-score for Class 1.0 (collaborationist) was only 0.76, with Recall = 0.65. A low Recall (35% of collaborationist messages missed) indicated mediocre performance for a critical class, which was a consequence of a small and non-representative training sample. The model was retrained on a significantly extended

corpus that included automatically labelled data, which enabled it to overcome the problem of training sample deficit (Fig. 6 and Tables 22–23).

Table 22
Model analysis

Parameter	Version 1,0 (Limited Sampling)	Version 2,0 (Advanced Sampling)
Scope of study	14,835 samples	141,357 samples
Distribution of classes	0.0: 72.5%, 1.0: 13.9%, -1.0: 13.6%	0.0: 50.6%, -1.0: 28.7%, 1.0: 20.7%
Architecture	Ensemble (MNB on TF-IDF Word + LR on Count Char)	
General Accuracy	0.9178	0.9438
ROC AUC (OvR)	-	0.9908
Log Loss	-	0.1960

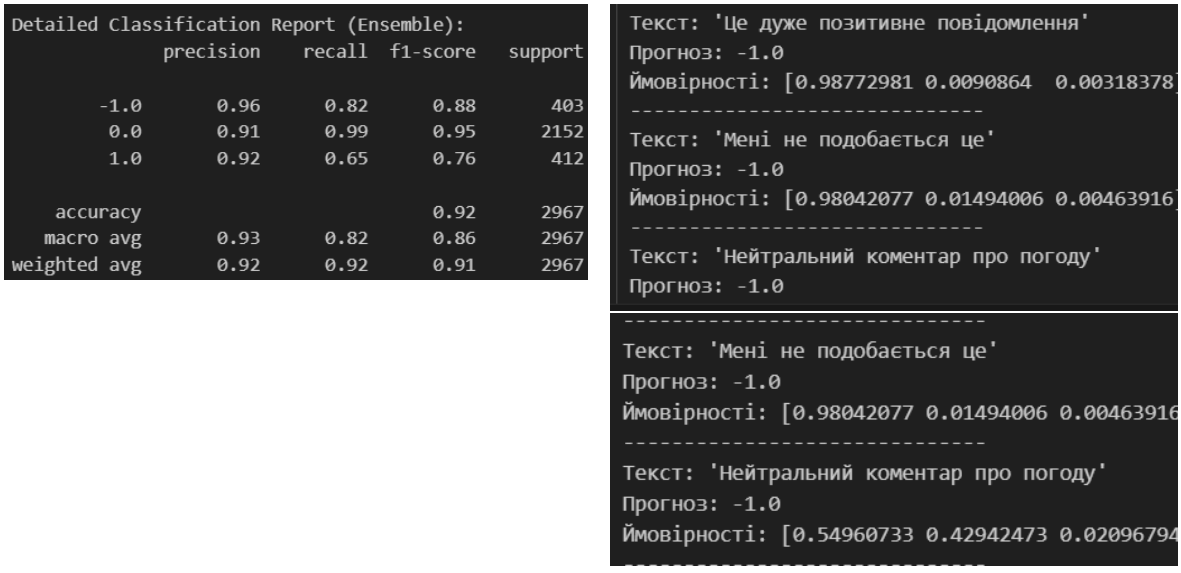

Figure 5: Evaluation of models and example of using the first model.

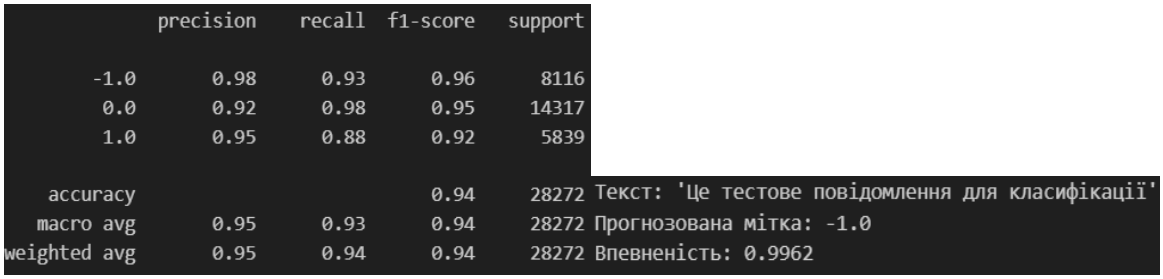

Figure 6: Evaluation of models and example of using the second model.

The model exhibits excellent discriminating ability (ROC AUC = 0.9908) and balanced accuracy (Balanced Accuracy = 0.9294). A high Precision (0.95) for a positive class is critical to minimising false positives before being applied at the account level. The model achieves excellent results, with an accuracy of 94.38% on a test set of 28.272 samples (Table 24, Figs. 7–9). It is a high indicator for the problem of three-class text classification. The balanced accuracy (92.94%) indicates that the model performs well, even with class imbalances. Cohen's Kappa (0.9079) and MCC (0.9091) confirm the high quality of the classification, excluding randomness. The ROC AUC (0.9908) indicates the model's excellent ability to distinguish between classes.

Table 23
Model Analysis – Detailed Class 1.0 (Collaborative) Metrics on a Test Sample (28,272 Samples)

Metric	Meaning	Analysis	Explanation
Precision	0.9538	High accuracy	If the model is classified as a "collaborator", the probability of correctness is ≈ 95%.
Recall	0.8808	Improved Search:	The model identifies ≈ 88% of genuine collaborationist messages.
F1-Score	0.9159	A significant improvement over version 1.0 (0.76).	

Table 24
Detailed evaluation of model metrics

Metrics	F1-Score	Precision	Recall	Supp	FP	FN	TP	TN	Value
Macro	0.9400	0.9527	0.9294						
Micro	0.9438	0.9438	0.9438						
Weighted	0.9436	0.9454	0.9438						
Class -1,0	0.9567	0.9849	0.9300	8116	116 (0.41%)	568 (2.01%)	7548	20040	
Class 0,0	0.9476	0.9196	0.9773	14317	1224 (4.33%)	325 (1.15%)	13992	12731	
Class 1,0	0.9159	0.9538	0.8808	5839	249 (0.88%)	696 (2.46%)	5143	22184	
								Accuracy	0.9438
								Balanced Accuracy	0.9294
								Cohen's Kappa	0.9079
								Matthews Correlation Coefficient	0.9091
								ROC AUC (One-vs-Rest):	0.9908
								Log Loss	0.1960
								Hamming Loss	0.0562

Analysis by classes (from the error matrix):

1. For the "-1.0" (negative) class, the model achieves the best precision (98.49%), rarely misclassifying texts as negative. Recall (93.00%) - misses 7% of truly negative texts, the most stable class for a model.
2. For the class "0.0" (neutral), the model achieves the highest recall (97.73%), indicating that it finds neutral texts well. Precision (91.96%) is lower due to some confusion with other classes, the largest class in the dataset (14,317 samples).
3. For the "1.0" (positive) class, the lowest recall (88.08%) is the most difficult to recognise. Precision (95.38%) is high – when a model says "positive", it is usually right for the smallest class in the dataset (5.839 samples).

The model makes relatively few mistakes:

1. The overall margin of error is only 5.62% (Hamming Loss).
2. Most recall issues for a positive class (11.92% False Negatives).
3. The Negative Class has the fewest False Positives (0.41%).

The model demonstrates stable performance across all classes, yielding powerful results for negative texts. The Log Loss (0,1960) indicates good probability calibration, which is crucial for practical applications.

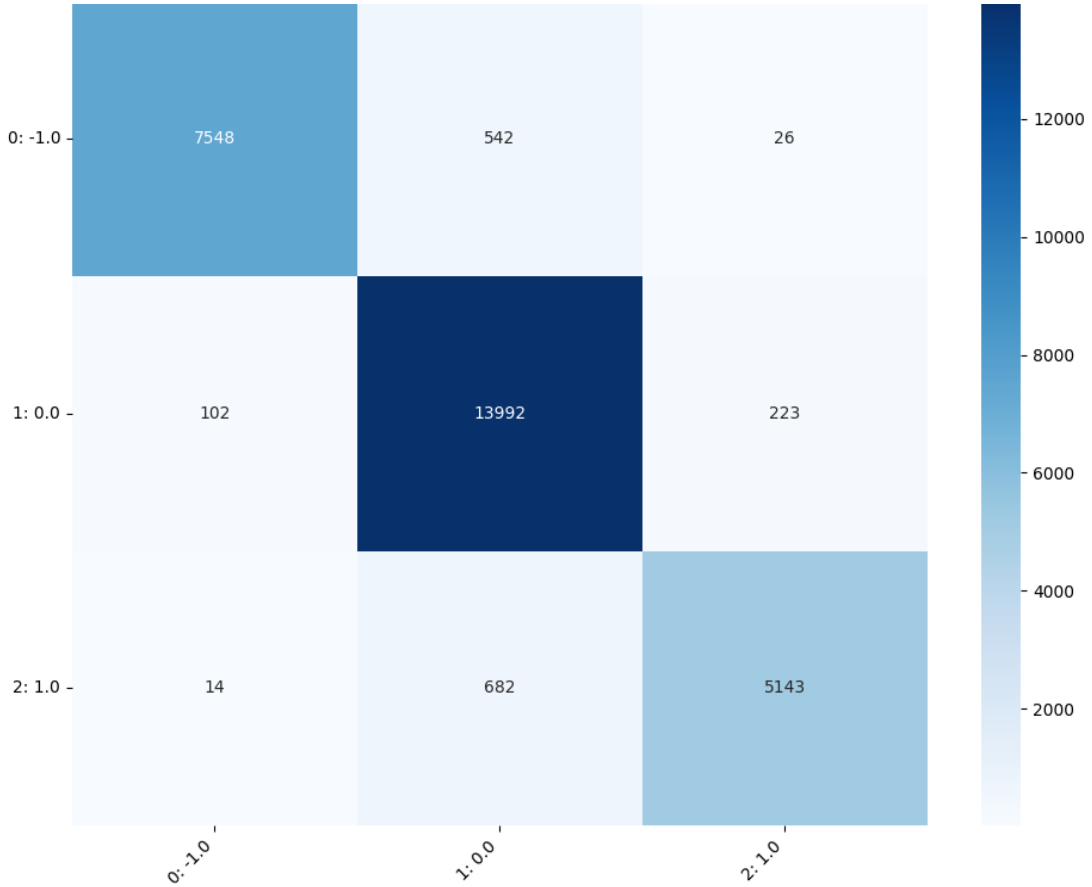

Figure 7: Confusion matrix, where on the left is the real class, and on the bottom is the predicted class.

Let's describe the stage of training the user account classification model. The purpose of the second model is to perform a binary classification of accounts into 0 (non-collaborator) and 1 (collaborator) based on aggregated content and metadata. Dataset size (372940, 7).

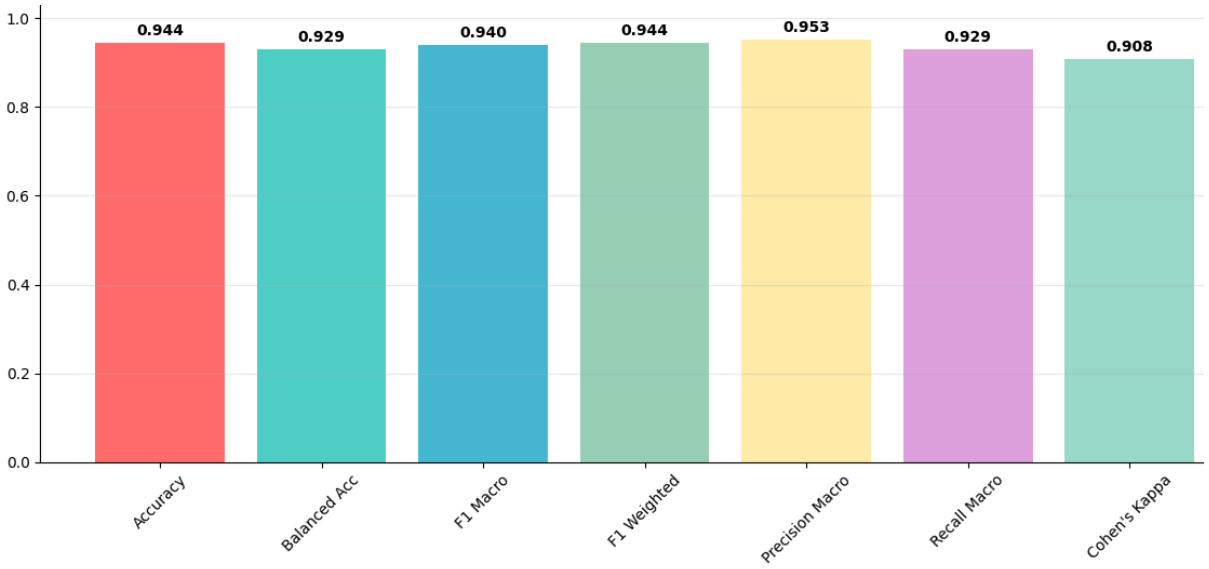

Figure 8: Model Metrics Comparison.

	precision	recall	f1-score	support
-1.0	0.98	0.93	0.96	8116
0.0	0.92	0.98	0.95	14317
1.0	0.95	0.88	0.92	5839
accuracy			0.94	28272
macro avg	0.95	0.93	0.94	28272
weighted avg	0.95	0.94	0.94	28272

Figure 9: Detailed classification report.

Data preparation and balance sheet (Fig. 10):

1. The volume of samples (authors) is 1,827 unique authors (combined data from the collaborationist and pro-Ukrainian groups).
2. Threshold definition: The author is classified as class 1 if the share of their collaborationist messages is 0.6% (threshold_ratio = 0.006).
3. Final class balance (authors): 1 (collaborators) 39.24% (717 authors); 0 (non-collaborators) 60.76% (1,110 contributors).
4. The class ratio is 1:1.55 (well-balanced dataset).

Example of a part of a dataset:

	Link to the author's ID	group	Date	predicted_label	Confidence
0	https://t.me/kherson_talk	1140052638	2025-05-02 20:49:23	0.0	0.957116
1	https://t.me/kherson_talk	1140052638	2025-05-02 20:48:12	0.0	0.788618
2	https://t.me/kherson_talk	1140052638	2025-05-02 20:46:07	0.0	0.966518
3	https://t.me/kherson_talk	1140052638	2025-05-02 20:42:06	0.0	0.906539
4	https://t.me/kherson_talk	1140052638	2025-05-02 20:37:10	0.0	0.900246

Class distribution for 0.0 – 348299, 1.0 – 20746 and -1.0 – 3895. The percentage of collaborationist messages is 4.52%. The total number of authors is 927. Authors with at least one collaborationist message 735. Classification of authors (threshold 0.6%): collaborators 717 and non-collaborators 210.

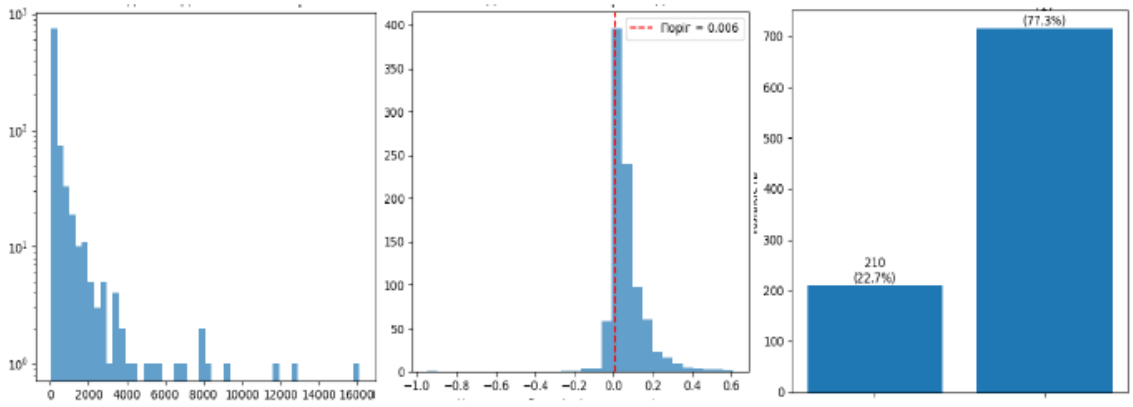

Figure 10: a) Distribution of messages by authors, where x is the number of messages per author and y is the number of authors; b) Distribution of the share of collaborative messages, where x is the share of such messages and y is the number of authors (threshold = 0.006); c) Distribution of author classes (0.6% threshold), where x is the author class and y is the number of authors, the first column is not a collaborator, the second is a collaborator.

After clearing the dataset of noise, new input data for training were obtained. The ratio of collaborators / non-collaborators is 0.646 (Fig. 11–14). Data for training: number of samples 1827, distribution of classes [1110 717], percentage of collaborators 39,24%. Detailed statistics on sources for the MAIN dataset: the number of authors is 927, collaborators 717, their collaborators 210 and the average share of collaborators. Messages 0.0646. Detailed statistics on sources for the ADDITIONAL dataset: number of authors 900, collaborators 0, non-collaborators: 900, average share of collaborators. Messages -1,0000. Dataset Balance Score: Class Ratio: 1:1.55, Dataset Well Balanced. The primary dataset is 372,940 messages from 927 authors. After merging with the pro-Ukrainian group, 412,531 messages from 1,827 authors. The minimum number of duplicates (9) indicates the quality of data collection. Only 4.52% of reports are classified as collaborationist (20,746 out of 372,940). It shows that open collaborationist rhetoric is a minority, even within the respective groups. Statistics on authors: 735 out of 927 authors (79%) have at least one collaborationist message. At a threshold of 0.6%, 717 authors are classified as collaborators (77.3%). Using the 0.6% threshold of collaborationist messages to classify the author. It is a conservative approach that allows you to take into account even minimal collaborationist activity towards unification, characterised by a strong imbalance: 77.3% of collaborators versus 22.7% of non-collaborators.

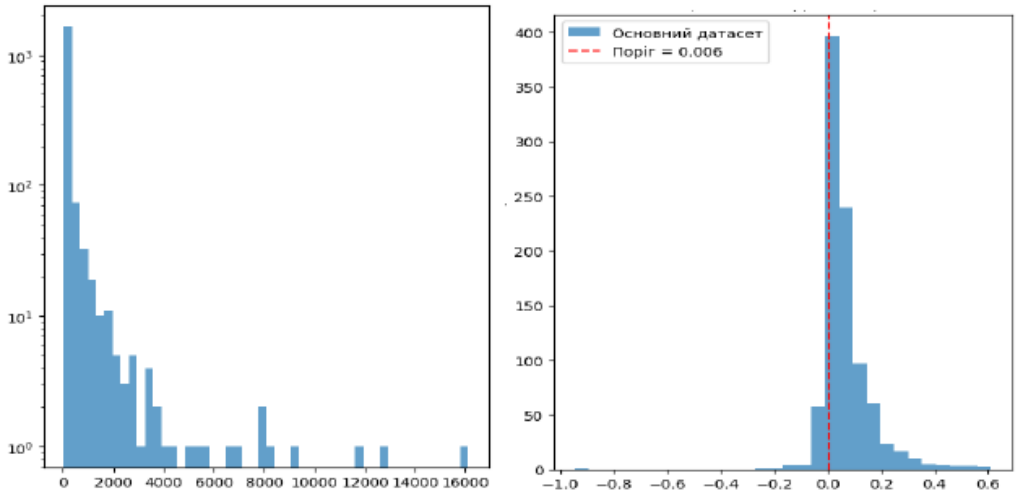

Figure 11: a) Distribution of messages by authors, where x is the number of messages per author and y is the number of authors; b) Distribution of the share of collaborative messages (primary dataset), where x is the share of such messages, and y is the number of authors (threshold = 0.006).

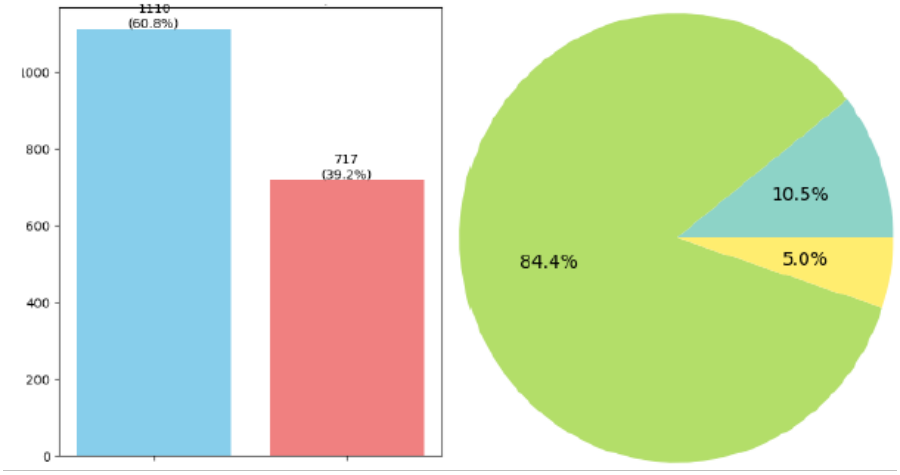

Figure 12: a) Distribution of classes of authors, where x is class and y is number, the first column is not a collaborator, the second is a collaborator; b) distribution of messages into classes, where green is 0, blue is -1 and yellow is 1.

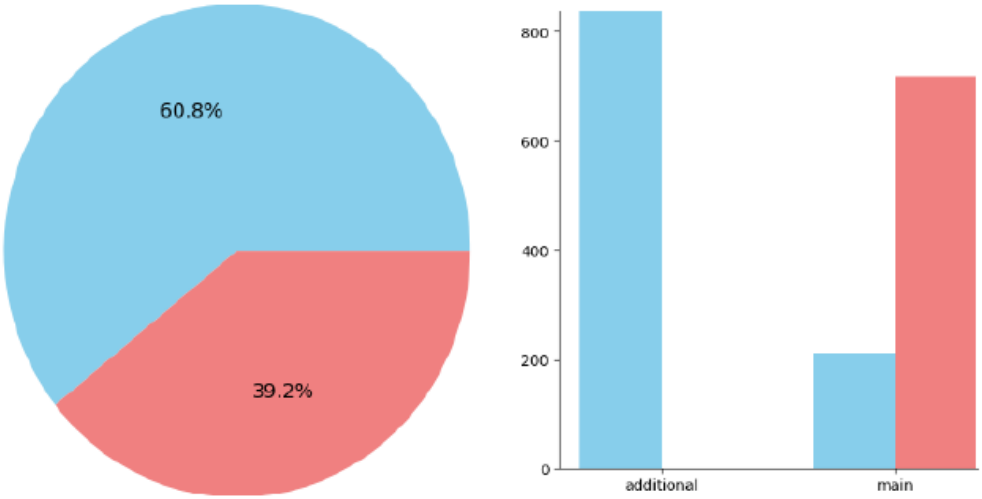

Figure 13: a) Distribution of authors into classes, where blue is not a collaborator, red is a collaborator; b) distribution of classes by sources, where x is the source and y is the number of authors.

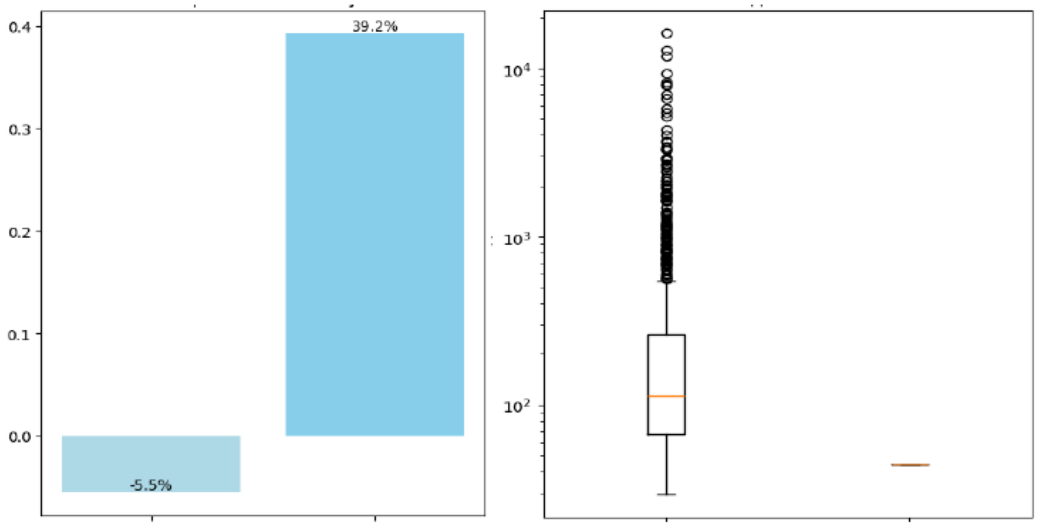

Figure 14: a) Comparison of the balance, where x is the class, and y is the proportion of the positive class, the first column is the message, the second is the authors; b) analysis by datasets, where x is the class and y is the number of messages, the first column is the base, the second is additional.

,After merging with the pro-Ukrainian dataset, the improved balance is 39.2% of collaborators vs. 60.8% of non-collaborators. A ratio of 1:1,55 is an acceptable level for machine learning.

Most authors have low activity (fewer than 1000 posts). A small number of hyperactive users. Typical power distribution for social networks. Precise bimodal distribution: authors either have a tiny proportion (about 0) or a significant one. It confirms the correctness of the threshold approach. The primary dataset from collaborationist groups may have bias. An additional pro-Ukrainian dataset helps, but does not eliminate the problem. The data is dated 2025, which may not reflect the evolution of views. Lack of temporal analysis of changes in the behaviour of the authors. Geographical reference: The data were collected from the collaborationist group in the Kherson region, which may not accurately represent the overall situation. Self-censorship or tactical behaviour of users is possible.

The models were trained on the TF-IDF matrix representing the authors' aggregated text (Tables 25–26, Figs. 15–16). The training sample consists of 1,461 samples, and the test sample comprises 366 samples. The distribution of classes in the educational sample is [888, 573]. The size of the TF-IDF matrix is (1461, 10000). All models (especially SVM and RF) demonstrated a Recall of 0.99 (99% detection of collaborators), which is a priority for security systems. The models are trained to aggressively detect Class 1, resulting in a relatively lower Precision (0.82–0.83) and, accordingly, a greater number of False Positives (falsely identified collaborators) – 29 out of 366 of test samples.

Table 25
Classification Report

Metric	Precision	Recall	F1-score	Support
Logistic Regression				
0	0.98	0.86	0.92	222
1	0.82	0.97	0.89	144
macro avg	0.90	0.92	0.91	366
weighted avg	0.92	0.91	0.91	366
Random Forest				
0	0.99	0.86	0.92	222
1	0.83	0.99	0.90	144
macro avg	0.91	0.93	0.91	366
weighted avg	0.93	0.91	0.91	366
SVM				
0	0.99	0.87	0.93	222
1	0.83	0.99	0.90	144
macro avg	0.91	0.93	0.91	366
weighted avg	0.93	0.92	0.92	366

Table 26
Comparison and selection of the algorithm (text features)

Model	Test AUC	Accuracy	Precision (Class 1)	Recall (Class 1)	CV AUC
SVM	0.9770	0.92	0.83	0.99	0.9839 (+/- 0.0061)
Random Forest	0.9769	0.91	0.83	0.99	0.9812 (+/- 0.0102)
Logistic Regression	0.9769	0.91	0.82	0.97	0.9809 (+/- 0.0101

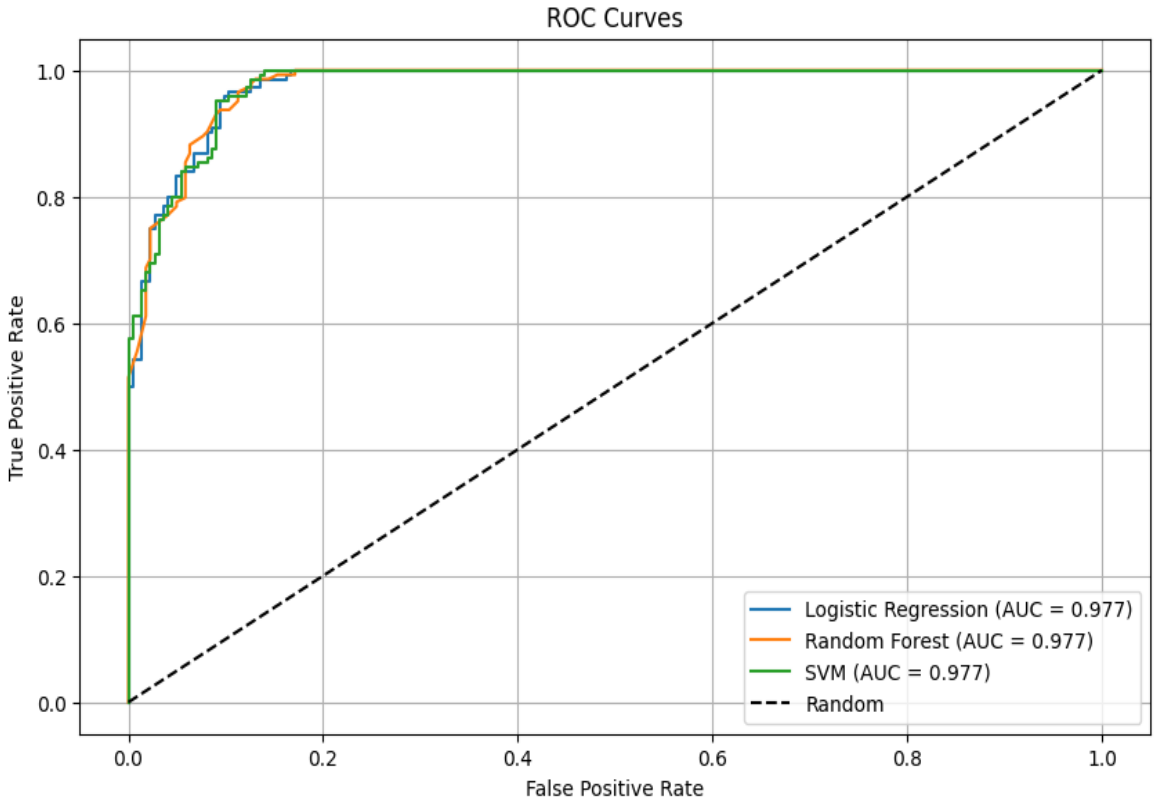

Figure 15: ROC curves.

The inclusion of numerical characteristics (activity, duration, and uniqueness of groups) in TF-IDF traits resulted in a slight decrease in performance. The combined model is AUC 0.9697. Textual features (TF-IDF) are the most dominant and informative for author classification. The SVM is chosen as the best model (AUC = 0.9770) due to its highest AUC and reliability, as confirmed by its low variability (CV AUC = 0.0061).

All three algorithms yield virtually identical results (AUC ~ 0.977). SVM is marginally better in accuracy (92% vs 91%). ROC curves show excellent discriminating ability – all models are far from a random line. Class 0 (non-collaborators) – very high precision (98–99%) – few false positives. Lower recall (86–87%) – 13–14% of real non-collaborators are missed. The model is conservative in its definition of non-collaborators. Class 1 (collaborators) – lower precision (82–83%) – 17–18% of false positives. High recall (97–99%) – finds almost all collaborators. The model is aggressive in finding collaborators. The model is configured for maximum detection of collaborators (high recall for class 1) and is ready to sacrifice precision in order not to miss potentially dangerous users. Error structure:

1. True Negatives 193 (correctly identified non-collaborators).

2. False Positives 29 (mistakenly labelled as collaborators).
3. False Negatives 2 (missed collaborators).
4. True Positives 142 (correctly found collaborators).

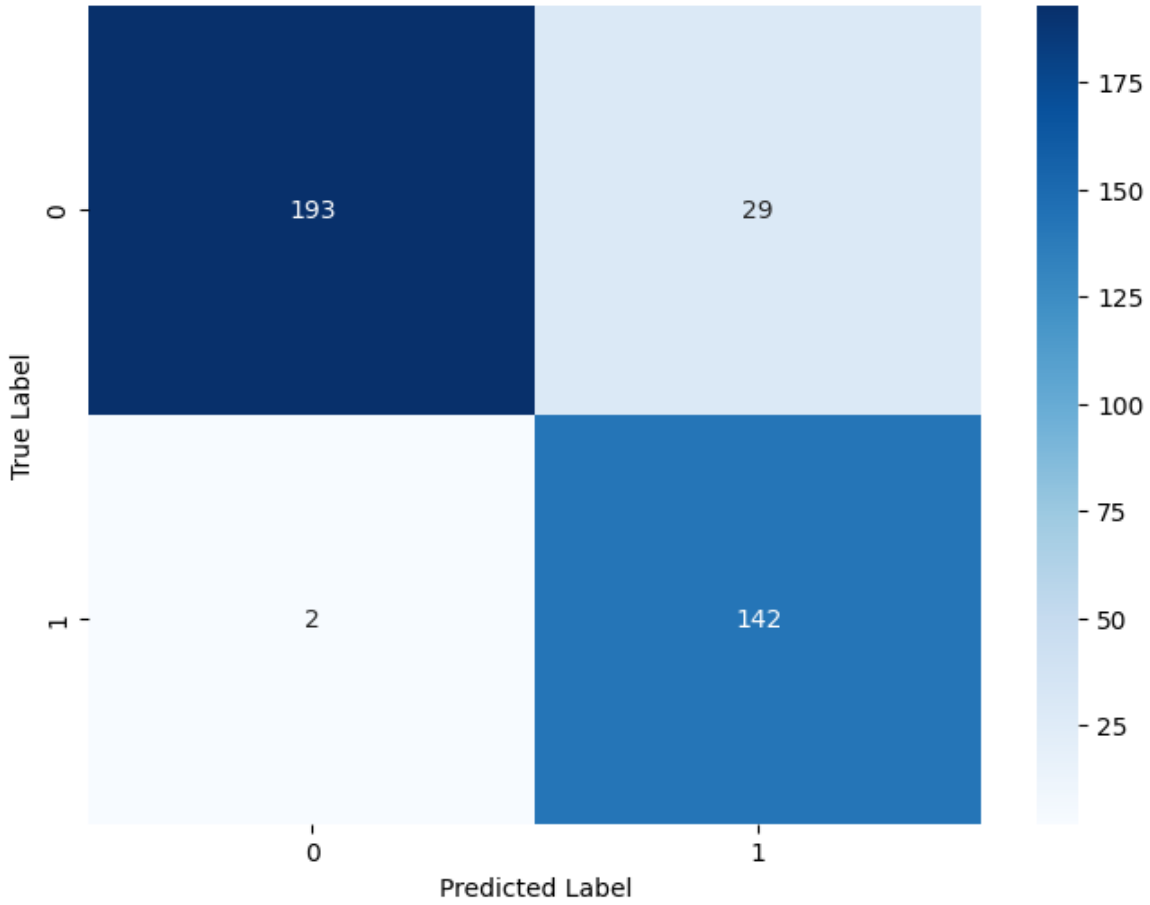

Figure 16: Confusion matrix for SVM.

Critical errors – only two missed collaborators (this is an excellent result for security) and 29 false positives (an acceptable level for further verification). The combined model has a slight deterioration:

1. The AUC decreased from 0.977 to 0.970.
2. It may indicate that additional signs do not provide significant improvement.
3. TF-IDF textual features are already informative enough.

Adequate size: 1,461 samples for training, 366 for testing. The training/testing ratio of ~80/20 is standard. The balance of classes in the test sample, with 222 and 144 participants (60.7% and 39.3%, respectively), is acceptable. A low standard deviation of the AUC (~0.01) indicates stability. SVM shows the least variability (±0.0061). The second model demonstrates excellent quality for practical applications. The high recall for collaborators (97–99%) makes it an effective tool for screening suspicious accounts. A small number of false negatives (2 out of 366) are critical for security tasks.

The high Accuracy (0.9438) and high precision (0.9538) for the positive grade provide quality input marking for the second model. The extremely high Recall (0.99) at an AUC of 0.9770 makes the model a highly effective tool for screening suspicious accounts while minimising the risk of False Negatives (collaborator omission). Based on the results obtained, both models demonstrate excellent quality and readiness for practical application, considering the Bias found in favour of Recall in the account classification model. Based on the training results of the SVM model (or Logistic Regression for the combined model), which demonstrated the highest discriminatory

ability (AUC 0.9770), an analysis of trait coefficients (weights) (TF-IDF terms) was carried out to determine their influence on the binary classification of the authors. Since the author classification model uses aggregated message text, these features reflect a stable lexical profile of the user.

The terms listed below in Table 27 received the highest positive coefficients (weights), making them key predictors of a user's belonging to Class 1 (Collaborator). Generally, this is propaganda vocabulary specific to Russian military and political rhetoric. The presence of these lexical units in the aggregated corpus of user messages is highly likely to indicate the acceptance or dissemination of propaganda narratives.

Table 27
Strong signs of collaborationist affiliation (Class 1)

Term/Phrase	Purpose
Khokhly	A derogatory name for Ukrainians (present in Russian propaganda).
The Armed Forces of Ukraine	It is often used to refer to the Armed Forces of Ukraine, but in a negative or derogatory context.
Putin	Political commentary centred around the Russian leader.
Svo	The abbreviation "Special Military Operation" is a key pro-Russian propaganda name for aggression.
Denazification	A key propaganda narrative justifying the invasion.
Crest	A singular version of the derogatory term.
Fascists	False accusation of aggression by the Ukrainian side.
Russia	Frequent mention of Russia in the context of its support or justification of actions.
NATO	Terms related to geopolitical conspiracy theories and justification for war.
American	References to the United States or the Western world as "aggressors".

The terms listed below in Table 28 received the most significant negative coefficients, indicating that they are key predictors of class 0 (non-collaborationist). These phrases often reflect patriotic, social, or neutral topics that are not related to pro-Russian rhetoric. These signs serve as a defence mechanism against false positives (False Positive, since their presence in the user's profile significantly reduces the likelihood of classification as a collaborator, than most common words. The model effectively utilises symbolic features (Counting n-grams in the first model) to capture variations in the spelling of these terms, ensuring the system's resistance to linguistic manipulation. The fact that the combined model (text with metadata) did not show a significant improvement compared to the model using text-only features indicates that the content of messages is more important than behavioural patterns (for example, the number of posts or the duration of activity).

Table 28
Strong signs of non-collaborationist affiliation (class 0)

Term/Phrase	Purpose
Glory to Ukraine	A direct patriotic slogan (a strong opposite sign).
Hello	A typical Ukrainian greeting that is absent in Russian-speaking collaborationist communities.
How are you	Neutral social phrases.
Money	Messages related to fundraising, financial assistance, and donations.
Help	Topics related to volunteering and social support.
The Armed Forces of Ukraine	The correct abbreviation of the Armed Forces of Ukraine (used in a positive/neutral context).
I	High frequency of using first-person singular pronouns in everyday, non-conflict contexts.
Thank you	Gratitude, which usually accompanies volunteer or social activity.
Weather	A typical example of neutral, social and household content.
Our	The use of the term "ours" in the context of the Ukrainian army or citizens.

To ensure complete transparency and identification of the limitations of the final account classification model (SVM on TF-IDF features), it is necessary to analyse examples of false positives (False Positives) and missed true cases (False Negatives) on the test sample (Table 29).

Recall that the final model was set to high Recall (detection of 99% of collaborators), which inevitably leads to an increase in False Positives. The system mistakenly classified the author as 1 (collaborator) when in fact he is 0 (non-collaborator).

Table 29
False Positive Error Analysis

Characteristics	Cause of occurrence
Quantity in the test	≈ 29 samples
Purpose of the error	Minimise False Negatives.
Main reason	High content of negative language/slang. An author who is a patriot may have used aggressive or obscene language to describe an enemy or situation. This vocabulary presumably coincides with the lexical patterns that the model associates with propaganda rhetoric (e.g., derogatory names for the opposite side).
Weaknesses of the model	The model relies on lexical coincidence without fully considering the context and position (Stance), which is opposite.

Hypothetical examples of aggregated text (False Positive): the author (patriot) who is marked as a Collaborator: "What an abomination this is, these [derogatory name], how dare they come to our land! Why the mistake: The terms "abomination" and "derogatory name" carry high weights in Class 1 due to their association with conflicting rhetoric, despite the overall patriotic context.

The system mistakenly classified the author as 0 (non-collaborator) when, in fact, it should have classified the author as 1 (collaborator). It is the most critical error for the security system (Table 30).

Table 30
False Negative Error Analysis

Characteristics	Cause of occurrence
Quantity in the test	≈ 2 samples (Lowest error rate!)
Purpose of the error	Identification of "disguised" collaborators.
Main reason	Mascuring tactics – the collaborator deliberately avoids using direct propaganda terms (e.g., svo, khokhly) in their messages, instead focusing on neutral, everyday, or highly contextual topics (e.g., weather, news).
Weaknesses of the model	The low frequency of key propaganda features in the aggregated text results in the total weight of the cumulative text falling below the threshold value, despite the text belonging to the positive class.

Hypothetical examples of aggregated text (False Negative): Author (collaborator) who is marked as Non-Collaborator: "Hey, how are you? It was quiet in our city [city name] today. Why the error: the text contains mostly neutral signs (hello, how are you, silent), which have a high weight in Class 0. There are no critical signs of Class 1.

The analysis of the error structure confirms that the model has successfully achieved its security goal:

1. Minimisation of critical errors – the number of False Negatives (missed collaborators) is only 2 out of 366 test samples. It confirms the high Recall (99%) and makes the system reliable for primary screening.
2. Tolerance to False Positive – the number of False Positives (29 samples) is acceptable, as these cases can be referred for manual expert verification (if P_Collaborator > 0.5 but P_Collaborator < 0.8, i.e. average confidence).

In general, the error structure of the model is consistent with its setting for maximum detection of a positive class (safety priority).

The use of traditional machine learning models (e.g. SVM on TF-IDF) to classify accounts, as we have seen, leads to high Recall (detection of collaborators) but has limitations in accuracy (Precision) due to insufficient understanding of the context. The integration of transformer models (e.g., BERT – Bidirectional Encoder Representations from Transformers) can significantly alter the error structure, particularly in the False Positive part. The main advantage of BERT is its ability to provide semantic and contextual representation (Semantic and Contextual Embeddings), as opposed to lexical weighting (TF-IDF):

Table 31
Mechanism for improving contextual understanding

Characteristics	TF–IDF/SVM	BERT
Understanding the signs	Lexical coincidence: takes into account the frequency and weight of individual words (for example, "abomination" or "ours").	Semantic connections: understands that "abomination" can refer to both the enemy (Patriot) and the authorities (Collaborator).
Problem	Polysemy/aggression, i.e. aggressive language of patriots, is mistakenly associated with propaganda → False Positive.	Contextual differentiation distinguishes between the tone and purpose of the message (e.g., anger directed at the enemy versus rage directed at one's authority).

If you replace the SVM/TF-IDF classifier with a BERT-based model, the following changes can be expected:

Table 32
Expected change in the error structure

Error	TF–IDF/SVM (Current Result)	BERT model forecast
False Positive	29 samples (caused by aggression, not related to collaboration).	A significant reduction – understanding the context will allow you to correctly classify aggressive but patriotic texts as Class 0 (non-collaborationist). It will increase the Precision for Class 1.
False Negative (missed collaborators)	2 samples (caused by masking/neutral content).	Reduction/no change possible – while BERT picks up hints better if the collaborator writes completely neutral text, BERT can also skip it. However, BERT can pick up on subtle stylistic cues that camouflage betrayal.

Table 33
Impact on key metrics

Metric	Current Value (SVM)	Expected change from BERT
Precision (Class 1)	0.83	Magnification (cl 0.90+) → fewer False Positives.
Recall (Class 1)	0.99	Maintaining a high level (target 0.95+) → to continue minimising False Negatives.
AUC	0.9770	Increase (target 0.98+) → better discriminatory ability.

Despite the advantages, the use of BERT imposes significant limitations that are especially important for practical deployment: computational costs and input length. Training and using

BERT requires significantly more GPU resources and time compared to SVM on TF-IDF. Standard BERT models have input sequence length limits (e.g., 512 tokens). Since the account classification model utilises aggregated text (which can be very long), specialised techniques must be employed (such as averaging, hierarchical models, or models with a larger context window, like LongFormer), which complicates the architecture. The introduction of BERT may be the next step in improving the quality and ethics of the classifier, particularly by reducing False Positives caused by the erroneous interpretation of aggressive pro-Ukrainian vocabulary. However, this will require significant investments in computational resources and further engineering optimisation of the model to work effectively with very long, aggregated texts.

9. Conclusions

Within the framework of this scientific project, a complete cycle of system development, from conceptualisation to experimental validation, has been implemented. The need to create a system for automated classification of collaborationist content in social networks, particularly Telegram, has been substantiated in response to the insufficient automation of monitoring information threats in groups associated with occupied territories. The scientific novelty, due to the lack of open, specialised datasets, has been emphasised, as well as systems for classifying collaborative content in the Ukrainian information space. The practical value lies in providing tools for security analysts, law enforcement agencies and social media moderators. Key problems have been identified, including the lack of specialised research on collaborators, the scarcity of training data in such projects, the ethical risks of false classifications, and the issue of ignoring/misclassifying pro-Ukrainian messages. Functional and non-functional requirements for both components of the system (classification of messages and accounts) are formulated, including hardware, software, and quality limitations.

Generated an initial dataset by collecting and manually labelling 10,861+ messages. Selected the TfidfVectorizer Ensemble Approach from Naïve Bayes and CountVectorizer (char-level) from Logistic Regression for message classification. Automatic labelling of more than 88,864 messages was performed. The final model, trained on 141,357 messages, achieved an accuracy of 0.9438 and an F1-score of 0.92 for the critical collaborationist class.

Based on the labelled body of 400,000 entries, a training sample was formed at the author's level. The optimal threshold for the author's classification was established: threshold_ratio = 0.006. Comparative training of Logistic Regression, Random Forest, and SVM with class weighting was conducted. The best SVM model achieved an AUC of approximately 0.977 and a Recall of roughly 0.99. A Classification Report revealed a strong bias in favour of Recall to ensure security.

The structure of errors (False Positives and False Negatives) is investigated, and the influence of linguistic features on the correctness of classification is analysed. Successful completion of control examples, including cross-domain persistence (e.g., Telegram, Twitter) and accurate distinction between patriotic and collaborationist content, is demonstrated. Recommendations for further improvement are formulated, including the use of SMOTE, further fine-tuning of thresholds, and collection of more data.

The control case in the study was used to independently verify the quality of the built system of automatic binary classification of user accounts based on their text messages. Its purpose was not only to assess the accuracy of the models but also to analyse the behaviour of classifiers in realistic conditions, closely approximating practical use. The control example, according to the methodology of the system, included:

1. Individual user messages processed using various TF-IDF vectorizers (word, char and mixed).
2. Generated model forecasts from a wide variety of classifiers: SVM (balanced, sensitive, manual weight), Logistic Regression (balanced, sensitive), Random Forest (balanced, sensitive), Gradient Boosting (standard and "sensitive" variant).

3. A set of metrics was used to compare models in the control example: accuracy, F1-macro, F1-weighted, F1-binary for class 1, F1 separately for classes 0 and 1, the number of positive predictions during testing, and the influence of the dynamic classification threshold.

Since identical data from the control set were used for each model, it became possible to objectively compare the quality of the algorithms, assess the stability of the models, and determine the best configuration for further use in the product.

The control example was formed in accordance with the general pipeline of the system:

1. Collection of texts from users who were not included in the training dataset.
2. Pre-linguistic processing, including tokenisation, noise cleaning, normalisation, and bringing to a single format.
3. Construction of three independent vector representations: TF-IDF word-level (unigrams and bigrams), TF-IDF char-level (3–5 characters in n-grams), TF-IDF mixed (combined use of word and char n-grams).
4. Passing control texts through all models trained in experiments.
5. Recording the results for each model, including the impact of sensitive thresholds (0.35 and 0.40).
6. Construction of summary tables of results, in particular, the top 10 models by F1-macro.

Thus, the control case functioned as a "final sanity check" – an independent test to test the model's ability to generalise data.

The analysis of the results showed a clear pattern:

1. SVM models with manual weights consistently demonstrate the highest quality on various vectorizers. Best Model: SVM and TF-IDF char with manual weight F1-macro = 0.9852, Accuracy = 0.9862.
2. Char-level vectorisation significantly improves results compared to word-level and mixed approaches, indicating resistance to spelling errors, the model's ability to capture stylistic patterns, and increased efficiency for short and informal messages.
3. Random Forest also performs well (F1-macro ≈ 0.97), but is inferior to SVM in terms of stability and accuracy.
4. Gradient Boosting consistently performs significantly worse than SVM and RF.
5. Logistic Regression gave the lowest quality, especially in sensitive mode.

The use of two thresholds (0.35 and 0.40) made it possible to assess how sensitive the model is to changes in the decision boundary:

1. A threshold of 0.40 ⟶ is more conservative, increasing precision class 1.
2. A threshold of 0.35 ⟶ is more aggressive, increasing the recall of class 1.

In general, changing the threshold has almost no effect on the best models (SVM/manual), which indicates their stability. All the best models predicted around 1270–1280 positive classes, which is consistent with the real balance of the data and suggests that:

1. Models are not subject to displacement.
2. Do not overestimate the dangerous class 1.
3. Do not demonstrate a "cautious" or "overstated" strategy.

It is vital for practical applications, such as identifying potentially dangerous accounts. According to all the metrics of the control example, the best model is SVM with manual class weight and char-level TF-IDF vectorisation. F1-macro = 0.9852; Accuracy = 0.9862;

F1 class 1 = 0.9813. It makes the model optimal for tasks such as account detection, risk assessment, and the detection of collaborationist patterns. Advantages of char-TF-IDF:

1. Symbols naturally model specific language constructions.
2. Work well in the Telegram environment (slang, errors, dialects).
3. Resistant to specially modified words (manual censorship).

Table 34
Comparison table of control case results

Control Result / Model	Dignity	Flaws	Limitations	Recommendations
SVM and char TF-IDF with manual weight (best combination)	Highest accuracy (Accuracy 0.9862), best F1-macro, resistance to threshold changes, and no overtraining	High data dimensionality, slower inference on large enclosures	Dependence on the quality of text cleaning	Use as the main model in the product
SVM and char TF-IDF with sensitive	High quality, recognises Class 1 well	Slightly lower macro-F1	Easily noticeable threshold sensitivity	Use as an alternative
SVM with word TF-IDF	Works well with standard text	Worse results in slang	Less error-resistant	Not recommended for Telegram
Random Forest (balanced)	High accuracy (F1-macro ≈ 0.97), stability	Exceeded inference time on large sets	Poorly scalable	Can be used as a fallback model
Random Forest (sensitive)	Boosts Class 1 recall	May overestimate positive predictions	Instability in short texts	Use in systems with high sensitivity
Gradient Boosting	Stable model, moderate inference time	Significantly lower metrics (around 0.91–0.92)	Needs regularization	Not recommended as the main
Logistic Regression	The fastest, easiest	Worst quality (F1-macro < 0.90)	Linearity of the model ⟶ weak generalizability	Can work in lightweight modes

Advantages of SVM (manual weight):

1. SVM works optimally in high-dimensional spaces.
2. Hand scales allow you to accurately adjust the penalty for class 1 errors.

Stable optimisation without strong overtraining.The control case confirmed the effectiveness of the proposed methodology for building a classification system for text accounts. The analysis of independent test results showed that the best models demonstrate high resistance to changes in input data, and are also able to maintain high classification accuracy even under challenging conditions (noise, slang, atypical vocabulary, short messages). The most effective configuration was the SVM model with hand scales combined with TF-IDF symbolic vectorisation, which yielded a significant performance increase compared to other approaches.

The work performed laid a solid scientific and technical foundation. Further development of the "Sphere" system provides for the expansion of its functionality and the integration of advanced methodologies:

1. Multilingual and cross-platform expansion, in particular, expanding support for other languages (English, Polish) and social networks (Twitter, VKontakte, Instagram) by adapting NLP pipelines to specific platform formats (length, media format, metadata).
2. Deep Learning/Transformers: Application of modern language models (BERT-based, XLM-R) to improve contextual understanding, which is critical for reducing False Positives caused by aggressive language; Research on the representation of messages in the form of graphs and the use of Graph Neural Networks to analyse the spread of misinformation and connections between accounts.
3. Improvements to the dataset and balancing, including the expansion of the manual labelling corpus through crowdsourcing or semi-automatic learning methods (Active Learning) and the use of generative models to synthesise new, representative examples for underrepresented classes.
4. Real-time operation and deployment: Development of a real-time monitoring system with integration of streaming APIs (Kafka) for prompt detection of "hot" topics; Creating a user interface (Dashboard) to visualise trends, indicators, and classification results.
5. Ethics and Interpretability (Explainable AI): Implementing XAI techniques to ensure transparency in decision-making and providing analysts with explanations as to why a particular profile was classified as a collaborator; Development of an MLOps pipeline for automatic retraining and monitoring of model quality degradation in a dynamic information environment.

The completed research provides a solid foundation for the transition to large-scale and high-precision solutions in the field of information security. Thus, the work performed has laid a solid foundation: from concept and analysis to the development, testing and statistical evaluation of models. Further work is aimed at expanding language and platform support, integrating modern DL-architectures, building real-time solutions, and ensuring the ethics and interpretation of the system.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] M. Nyzova, V. Vysotska, L. Chyrun, Z. Hu, Y. Ushenko, D. Uhryn, Smart tool for text content analysis to identify key propaganda narratives and disinformation in news based on NLP and machine learning, IJCNIS 17 (4) (2025) 113–175. doi:10.5815/ijcnis.2025.04.08.

[2] About PHEME, 2025. URL: https://www.pheme.eu/

[3] M. Cheng, S. Nazarian, P. Bogdan, VROC: variational autoencoder-aided multi-task rumor classifier based on text, in: Proceedings of the Web Conference 2020, WWW '20, Association for Computing Machinery, New York, NY, 2020, pp. 2892–2898. doi:10.1145/3366423.3380054.

[4] S. Jendoubi, A. Martin, L. Liétard, B. Ben Yaghlane, Classification of message spreading in a heterogeneous social network, in: Proceedings of the 15th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU '2014, Springer, Cham, Switzerland, 2014, pp. 66–75. doi:10.1007/978-3-319-08855-6_8.
[5] StopRU, 2023. URL: https://stopru.in.ua/
[6] RuFilter (Chrome extension), 2025. URL: https://chromewebstore.google.com/detail/rufilter
[7] V. Vysotska, A. Starchenko, L. Chyrun, Z. Hu, Y. Ushenko, D. Uhryn, Sentiment analysing and visualising public opinion on political figures across YouTube and Twitter using NLP and machine learning, IJIGSP 17 (5) (2025) 117–164. doi:10.5815/ijigsp.2025.05.08.
[8] V. Vysotska, S. Popp, V. Bulatova, Z. Hu, Y. Ushenko, D. Uhryn, Smart tool for identifying misinformation spread sources and routes in social networks based on NLP and machine learning, IJCNIS 17 (5) (2025) 114–165. doi:10.5815/ijcnis.2025.05.08.
[9] V. Vysotska, D. Ptushkin, R. Fedchuk, R. Lynnyk, Probabilistic thematic modelling of Ukrainian-language texts based on the latent Dirichlet allocation algorithm, in: Proceedings of the International Workshop on Applied Intelligent Security Systems in Law Enforcement, AISSLE '2025, CEUR Workshop Proceedings, Aachen, Germany, 2025, pp, 226–291.
[10] V. Vysotska, M. Yatsyshyn, R. Romanchuk, V. Danylyk, Information technology for automatic detection of calls for terrorist acts in social media posts and comments based on machine learning, in: Proceedings of the International Workshop on Applied Intelligent Security Systems in Law Enforcement, AISSLE '2025, CEUR Workshop Proceedings, Aachen, Germany, pp. 306–360.
[11] H. Padalko, V. Chomko, S. Yakovlev, P.P. Morita, Classification of disinformation in hybrid warfare: an application of XLNet during Russia's war against Ukraine, Radioelectron. Comput. Syst. 2024 (4) (2024) 46–58. doi:10.32620/reks.2024.4.04.
[12] V. Solopova, O.-I. Popescu, C. Benzmüller, T. Landgraf, Automated multilingual detection of pro-Kremlin propaganda in newspapers and Telegram posts, Datenbank Spektrum 23 (2023) 5–14. doi:10.1007/s13222-023-00437-2.
[13] F. Sufi, Social media analytics on Russia–Ukraine cyber war with natural language processing: perspectives and challenges, Information 14 (9) (2023). doi:10.3390/info14090485.
[14] C. Y. Park, J. Mendelsohn, A. Field, Y. Tsvetkov, Challenges and opportunities in information manipulation detection: an examination of wartime Russian media, arXiv preprint arXiv:2205.12382 (2022). doi:10.48550/arXiv.2205.12382.
[15] I. A. Pilkevych, D. L. Fedorchuk, M. P. Romanchuk, O. M. Naumchak, Approach to fake news detection using graph neural networks, J. Edge Comput. 2 (1) (2023) 24–36. doi:10.55056/jec.592.
[16] K. Durani, A. Eckhardt, W. Durani, T. Kollmer, N. Augustin, Visual audience gatekeeping on social media platforms: a critical investigation on visual information diffusion before and during the Russo–Ukrainian War, Inf. Syst. J. 34 (2) (2024) 415–468. doi:10.1111/isj.12483.
[17] C. Maathuis, I. Kerkhof, First six months of war from Ukrainian topic and sentiment analysis, in: Proceedings of the 10th European Conference on Social Media, ECSM '2023, Academic Conferences International Limited, Kidmore End, UK, 2023, pp. 163–173. doi:10.34190/ecsm.10.1.1147.
[18] R. Marigliano, L.H.X. Ng, K.M. Carley, Analyzing digital propaganda and conflict rhetoric: a study on Russia's bot-driven campaigns and counter-narratives during the Ukraine crisis, Soc. Netw. Anal. Min. 14 (1) (2024). doi:10.1007/s13278-024-01322-w.
[19] J. C. Aguerri, M. Santisteban, F. Miró-Llinares, The fight against disinformation and its consequences: measuring the impact of "Russia state-affiliated media" on Twitter, Crime Sci. 13 (1) (2024). doi:10.1186/s40163-024-00215-9.
[20] K. Lipianina-Honcharenko, Y. Bodyanskiy, N. Kustra, A. Ivasechko, OLTW-TEC: online learning with sliding windows for text classifier ensembles, Front. Artif. Intell. 7 (2024). doi:10.3389/frai.2024.1401126.

[21] M. Makhortykh, M. Sydorova, A. Baghumyan, V. Vziatysheva, E. Kuznetsova, Stochastic lies: how LLM-powered chatbots deal with Russian disinformation about the war in Ukraine, Harvard Kennedy School Misinformation Review (2024). doi:10.37016/mr-2020-154.

[22] B. Shultz, An entity-aware approach to logical fallacy detection in Kremlin social media content, in: Proceedings of the 2023 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM '23, Association for Computing Machinery, New York, NY, 2023, pp. 780–783. doi:10.1145/3625007.3627988.

[23] C. Maathuis, C. de Ridder, S. Stuurman, Analyzing the role of Ukrainian and Russian diaspora in disinformation campaigns, in: Proceedings of the 10th European Conference on Social Media, ECSM '2023, Academic Conferences International Limited, Kidmore End, UK, 2023. pp. 153–162. doi:10.34190/ecsm.10.1.1118.

[24] Sternenko, Telegram channel, 2025. URL: https://t.me/spilnotass