A multilingual analysis for political stance and manipulative language in Reddit comments on the Ukraine war

Victoria Vysotska[1,†], Alla Kuzko[1,†], Lyubomyr Chyrun[1,2,†], Yuriy Ushenko[3,†], Roman Romanchuk[1,*,†], Roman Lynnyk[1,†], Yaroslav Pelekh[1,†], Laura Duisembaueva[4,†] and Mariia Brygadyr[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 Fedkovych Chernivtsi National University, Kotsiubynskoho Street 2, 58012 Chernivtsi, Ukraine*
[4] *Al-Farabi Kazakh National University, Al-Farabi Ave, 71, 050040 Ust-Kamenogorsk, The Republic of Kazakhstan*
[5] *West Ukrainian National University, Lvivska Street 11, 46004 Ternopil, Ukraine*

Abstract

The article presents a comprehensive study of online discourse on Russia's war against Ukraine on the Reddit platform using natural language processing (NLP) techniques. A multilingual information system is proposed, which collects, cleanses, and analyses the sentiment of comments in English, Ukrainian, and Russian through thematic clustering. Particular attention is paid to the definition of a political position (stance detection): pro-Ukrainian, pro-Russian or neutral. The model is based on the multilingual embeder multilingual-e5-base, complemented by manual data markup and specialised dictionaries of manipulative and propaganda vocabulary. The paper also provides a comparative analysis of modern NLP solutions (Perspective API, MonkeyLearn, Watson NLU, BERT architectures). It substantiates the advantages of the proposed approach in the context of information security. The results of the study demonstrate the effectiveness of the system in identifying information narratives, dominant topics of discussion and the structure of users' emotional reactions. The created system can be used to monitor public sentiment, detect propaganda and analyse information operations in social networks.

Keywords

Reddit, NLP, multilingual classification, stance detection, text sentiment, information warfare, propaganda, manipulative vocabulary, text vectorisation, thematic clustering, multilingual-e5-base, BERT, public opinion, disinformation.

1. Introduction

In today's world, social media has become the main arena of the struggle for public opinion. In the context of Russia's war against Ukraine, the information space is a critical element of hybrid warfare. In particular, the Reddit platform, which is characterised by an active and global English-speaking community, has become one of the tools for shaping the views of an international audience. Thousands of comments related to events in Ukraine are published on Reddit every day. These statements are a direct reflection of public sentiment, encompassing emotions, support, criticism, and deliberate misinformation [1–3]. The analysis of these comments enables us to

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); alla.kuzko.sa.2022@lpnu.ua (A. Kuzko); lyubomyr.v.chyrun@lpnu.ua (L. Chyrun); y.ushenko@chnu.edu.ua (Y. Ushenko); roman.v.romanchuk@lpnu.ua (R. Romanchuk); roman.o.lynnyk@lpnu.ua (R. Lynnyk); yaroslav.m.pelekh@lpnu.ua (Y. Pelekh); duisembaevalaura@gmail.com (L. Duisembaueva); m.bryhadyr@wunu.edu.ua (M. Brygadyr)
0000-0001-6417-3689 (V. Vysotska); 0000-0002-6356-4992 (A. Kuzko); 0000-0002-9448-1751 (L. Chyrun); 0009-0002-7620-5355 (Y. Ushenko); 0009-0004-4352-1073 (R. Romanchuk); 0009-0007-0948-4338 (R. Lynnyk); 0000-0002-4339-8093 (Y. Pelekh); 0009-0005-5402-6753 (L. Duisembaueva); 0000-0002-1101-7479 (M. Brygadyr)
© 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/paper21.pdf

understand precisely how the image of war is perceived by a global audience [3–6]. The relevance of this topic is also due to [6–12]:

1. The rapid spread of fake news, memes, and propaganda clichés needs to be identified and addressed.
2. Insufficient study of Reddit as a source of data for the Ukrainian context.
3. A need for prompt monitoring of public sentiment outside Ukraine.
4. The availability of large data sets for NLP research.

Thus, the combination of comment sentiment analysis and the detection of manipulative vocabulary on Reddit is not only relevant but also a socially significant area of research in computational linguistics.

The purpose of this study is to develop an information technology for a comprehensive analysis of comments in different languages on Reddit related to Russia's war against Ukraine, using natural language processing methods. It is envisioned to identify the emotional colouring of statements (tonality) and the frequent use of vocabulary, which may indicate manipulative influence. It will enable us not only to establish the prevailing public emotions but also to identify the structural features of information campaigns and propaganda narratives. Research objectives:

1. Download and prepare a selection of Reddit comments on the war in Ukraine.
2. Clean up texts (remove noise, duplications, unwanted characters).
3. Conduct a preliminary analysis of the comment language and determine the comment languages to distribute into groups.
4. Apply sentiment analysis tools (TextBlob, VADER, or others) to classify sentiment.
5. Build a distribution of comments by tone.
6. Highlight negative and neutral comments for more detailed analysis.
7. Conduct lexical analysis and build a TF-IDF model to identify key terms.
8. Visualise common vocabulary (wordcloud).
9. Form a dictionary of potentially manipulative vocabulary.
10. Analyse examples of comments containing propaganda frames or disinformation narratives.

The object of the study is the process of forming online discourse around Russia's war against Ukraine on English-language social media platforms (including Reddit), as well as the dissemination of manipulative and emotionally colored messages among users. The subject is the methods and means of analysing text statements (comments) from Reddit users that contain emotionally charged or potentially manipulative language related to Russia's war against Ukraine. The presence of typical propaganda narratives ("denazification", "liberation", "NATO provocation", etc.) is also investigated. The scientific novelty of the study is reflected in the following:

1. Using Reddit as a source to research information warfare in the context of Ukraine.
2. combination of classical methods of sentiment analysis with thematic and frame analysis of manipulations.
3. Identifying keywords and patterns used for propaganda.
4. Creation of a frequency dictionary with examples of manipulative vocabulary.
5. Approbation of NLP methods on the corpus of informal English-language online commentaries.

The results of the study can be used [6–12]:

1. To create automated systems for monitoring public sentiment.
2. To detect new waves of information attack or propaganda.
3. In the work of state think tanks, journalists, and public organisations.
4. To train models for detecting disinformation and bot activity.

5. As part of an interactive public opinion dashboard with visualisation elements (e.g. via Streamlit or Gradio).

For the study, a relevant, interdisciplinary topic was chosen that combines computational linguistics, analysis of public sentiment, elements of information security, and NLP. The purpose, objectives, scientific novelty and practical value of the project are detailed. The results can be helpful in both academic and applied environments for understanding the influence of information in wartime.

2. Related works and comparative analysis of similar solutions

In the context of modern information warfare, a crucial task is to automatically identify users' political positions based on their comments [12–18]. Comments on platforms like Reddit may contain signs of manipulative language, toxicity, or overt political leanings. There is a need to create systems capable of detecting neutral, pro-Russian or pro-Ukrainian positions [18–21]. The urgency of the problem stems from the need to counter disinformation, combat fake news, and enhance information security [21–30]. The modern landscape of natural language processing tools offers a wide range of solutions for analysing textual content; however, the vast majority of them focus on traditional tasks, such as determining emotional tone, detecting toxicity, or performing general sentiment analysis. The specific task of defining a political position, particularly in the context of the armed conflict in Ukraine, remains a largely untapped area, creating unique opportunities for the development of specialised technological solutions. Automatic identification of a political position is an interdisciplinary task that combines approaches from the fields of machine learning, natural language processing (NLP), sociolinguistics, and media studies [1–12]. Let us review and analyse the leading publications that have influenced the development of these approaches [13–20]. In [13], the authors present a detailed methodology for analysing position in social networks, highlighting the difference between emotional assessment (sentiment) and position on a particular topic. The study [14] is devoted to fact-checking with an emphasis on contextual verification of statements, accompanied by the analysis of rhetorical techniques. The work [15] is an overview of methods for detecting disinformation, including classification, graph approaches, and argumentation models. The authors in [16] propose an explainable NLP that explains the reasons why a statement is deemed true or false. In [17] the authors develop models for detecting propaganda, hostile messages, and ideologically colored statements. The work [18] provides an overview of classical and modern methods for sentiment analysis, including rule-based, statistical, and deep learning approaches. A study [19] describing the implementation of the BERT architecture was a revolutionary step in NLP and provided a basis for further training in stance detection problems. In [20], an overview of the use of transformers in social analysis problems, including hate speech, misinformation, and stance classification, is carried out. Modern methods include [21–30]:

1. Using BERT, DistilBERT for stance classification (with fine-tuning).
2. Transformers with a special position token ("[STANCE]").
3. Multilingual models (XLM-RoBERTa) – especially relevant for multilingual data (e.g. English-Ukrainian-Russian contexts).
4. Use of unsupervised approaches (topic modelling, clustering) for preliminary analysis.
5. Use of manipulative or propagandistic vocabulary in dictionaries.

We will carry out a detailed review of existing approaches and their functional limitations.

Google Jigsaw developed the Perspective API as a cutting-edge machine learning tool designed to automatically identify problematic content in online discussions. The system is focused on identifying a wide range of negative manifestations in texts, which makes it popular among social platform moderators. The system is capable of evaluating a variety of text characteristics, including

overall toxicity, insult, profanity, threats, identity-based attacks, and severe toxicity. The API offers a user-friendly interface for seamless integration with various platforms and research projects, making it accessible to a broad range of users. Critical shortcomings for political analysis:

1. Lack of understanding of the political context.
2. Contextual errors in estimation.
3. Language and cultural barriers.
4. Western-centricity of algorithms.

The Perspective API is not fundamentally intended to analyse the political positions of participants in discussions. Instead of determining whether the user supports a particular side of the conflict, the system only assesses formal signs of aggressiveness or resentment, which can lead to serious interpretation errors. Real-world examples show that phrases like "I support the heroic defence of Ukraine" can get high toxicity ratings simply because of mentioning military actions, despite their obviously positive nature in the context of supporting the victim of aggression. The system demonstrates a critically low quality of work in the Ukrainian and Russian languages, which is manifested in incorrect recognition of Cyrillic characters, tokenisation errors, and a misunderstanding of culturally specific expressions and idioms. The educational data of the system are primarily based on English-language content from Western countries, which makes it unsuitable for an accurate assessment of the specifics of the post-Soviet information space, with its unique forms of political discourse.

Leading commercial platforms such as MonkeyLearn, IBM Watson Natural Language Understanding, and Amazon Comprehend are comprehensive solutions for the enterprise segment. These systems provide ready-to-use tools for a wide range of text processing tasks, from analysing customer feedback to automatically categorising documents. Functionalities of leading platforms:

1. MonkeyLearn stands out for its customisation flexibility, allowing users to train their own classifiers for specific tasks, conduct multivariate sentiment analysis, and automatically detect topic categories in large arrays of text.
2. IBM Watson NLU offers a comprehensive approach to analysis, including the extraction of emotional markers, conceptual connections, and named entities, and also provides advanced means of visualising results for business intelligence.
3. Amazon Comprehend specialises in automatically analysing large documents, efficiently identifying text language, key phrases, dominant topics, and emotional characteristics without requiring pre-tuning.

System limitations for research purposes:

1. The problem of the "black box".
2. Limited control over learning.
3. Economic barriers.
4. Inferior multilingualism.
5. Lack of political categories.

The closed nature of the algorithms used by these platforms makes it impossible to gain a deep understanding of the decision-making processes, which is critical for scientific research. Users do not have the opportunity to fine-tune the systems to the specific needs of the Ukrainian-Russian political discourse. Procedures for further training models on their own data are either inaccessible or highly complex and costly, making them impractical for academic or volunteer projects. The cost of processing large amounts of social media (hundreds of thousands or millions of comments) through the APIs of these services can become prohibitive for research budgets, especially in the context of long-term monitoring. Despite the declared support for multiple languages, the quality of work with Ukrainian and Russian is significantly inferior to that of English, as evidenced by

incorrect recognition of specific vocabulary, misunderstanding of synonymous series, and errors in processing morphologically complex constructions. None of the platforms offers ready-made classifiers for determining a political position, such as "pro-Ukrainian" or "pro-Russian", which requires the creation of entirely custom solutions on top of the basic functionality.

A family of transformer models, including BERT, RoBERTa, XLM-RoBERTa, and DistilBERT, has revolutionised the field of natural language processing by offering a fundamentally new approach to understanding context in texts. These architectures have become the foundation for most advanced NLP systems due to their ability for deep contextual analysis. Technical characteristics of key models:

1. BERT (Bidirectional Encoder Representations from Transformers) has implemented the revolutionary principle of bidirectional context analysis, which enables the model to consider both previous and subsequent words when interpreting each element of the text.
2. RoBERTa introduces an optimised version of BERT with improved learning strategies, resulting in higher accuracy on most benchmarks while maintaining architectural compatibility.
3. XLM-RoBERTa extends the capabilities of the base model to more than a hundred languages, theoretically providing multilingual functionality for global applications.
4. DistilBERT offers a compromise solution between performance and precision, providing significantly faster machining at the expense of some reduction in result quality.

Challenges of practical application:

1. There is a need for specialised additional training.
2. The problem of the interpretation of decisions.
3. Superficial contextual analysis.
4. Language imbalances in educational data.
5. High requirements for computing resources.

Basic transformer models lack a built-in understanding of the concept of political position. After downloading from repositories such as Hugging Face, they require painstaking additional training on specially prepared and marked data, which requires significant technical resources and expertise. Transformer models function as complex "black boxes", which makes it challenging to understand the logic of their solutions without using specialised explanation methods such as LIME or SHAP, which are resource-intensive in themselves. Without the integration of thematic dictionaries or specialised analysis modules, models can classify texts according to formal patterns, without understanding the deep meaning of key terms such as "de-occupation", "collaboration", and "militarisation" in the Ukrainian context. Even the most advanced multilingual models, such as XLM-RoBERTa, demonstrate uneven quality of work across different languages due to an imbalance in educational resources, where Ukrainian is significantly underrepresented compared to English. Effective retrotraining and application of transformer models on large datasets requires access to powerful GPUs or cloud computing platforms, which may not be available for many research projects.

Specialised research initiatives:

1. The TWEETSBK (Knowledge Base for Political Tweets) project represented an ambitious attempt to create a centralised knowledge base for classifying politically oriented messages on Twitter. The developers attempted to systematise the patterns of political discourse and develop universal tools for recognising them. The study focused exclusively on the American political context and English-language data, making its results of little use for analysing other regional conflicts or multilingual environments.

2. The PHEME project (dataset and position detection on rumours) proposed interesting methodological approaches for analysing how users support or refute circulating claims on social networks, which have some similarities with stance detection tasks. Despite the conceptual proximity, the methods of this project are not fully adapted for the comprehensive identification of political positions in multifaceted conflicts, particularly without considering the specific regional context.
3. Explanatory NLP models (LIME, SHAP with BERT) – the integration of machine learning explanation methods with transformer architectures opens up opportunities for understanding the logic of decision-making by models, which is especially important for political analysis tasks. The implementation of such systems requires significant computing resources and deep technical expertise; however, there are no standardised solutions specifically adapted to the task of stance detection in the political context.

3. Statement of the research task

The central problem our system addresses is the critical lack of reliable tools to automatically determine the political stance of social media users in the context of the Russian-Ukrainian conflict. Existing methods on the market are unable to fully identify and classify pro-Russian or pro-Ukrainian narratives, especially when they are presented in complex, multi-layered or deliberately veiled formulations. The main goal of the project is to create a comprehensive tool that is capable :

1. Accurately recognise a political position based on the textual content of comments, regardless of their stylistic design.
2. Conduct an in-depth analysis of the thematic structure and lexical composition of the texts under study.
3. Efficiently handle multilingual content, including English, Ukrainian, and Russian.
4. Adapt to new types of content through additional training mechanisms using fresh examples.
5. Ensure transparency of classification through detailed analysis of the most significant words and phrases.

The development faces a number of serious challenges:

1. Linguistic diversity of online communication.
2. Technical limitations of the platforms.
3. The problem of artificial content.
4. The complexity of political discourse.

Social networks are characterised by an extremely high level of linguistic variability, including transliteration, informal abbreviations, dialectisms, and new forms of slang that are constantly evolving. The APIs of most social platforms have significant limitations on data collection, which makes it challenging to form representative samples for analysis. The widespread use of bots and automated systems for content generation presents substantial challenges to the accurate analysis of public opinion. The high level of irony, sarcasm, and manipulative techniques in political discussions makes automatic interpretation a challenging task even for the most advanced algorithms. To overcome these challenges, it is necessary to implement an integrated approach: pre-processing improvements, vectorisation optimisation and development of analytical dashboards. Development of specialised algorithms for text cleaning, normalisation of various writing forms, automatic error correction, and creation of slang and colloquial vocabulary dictionaries for each supported language. Maximise the use of multilingual-e5-base capabilities to ensure high-quality representation of multilingual texts in vector space, taking into account cultural and contextual features. Creation of interactive visualisation tools that will allow you to

compare the distribution of political positions according to various criteria: thematic categories, social platforms, time periods, and demographic characteristics of users. Advantages of our approach (in the context of the disadvantages mentioned above):

1. Focus on stance detection in political discourse.
2. Multilingual support thanks to multilingual-e5-base.
3. Manual data markup is the basis of accuracy.
4. Thematic and lexical analysis is not only a matter of classification.
5. Flexible architecture, core open to additional learning.
6. A balanced combination of statistical and linguistic methods.

The system being developed is fundamentally different from traditional tonality analysis tools in that it focuses on identifying the user's political stance regarding the Russia-Ukraine conflict. Instead of simply determining the positive or negative colour of the text, the system classifies comments into three categories: pro-Ukrainian, pro-Russian, and neutral. Such specialisation proves to be critically important when analysing the information space in wartime, where a seemingly positive comment may actually contain manipulative or disinformation messages.

The use of the multilingual-e5-base model enables our system to efficiently work with texts in Ukrainian, Russian, and English simultaneously. This solution provides comprehensive coverage of the entire range of comments found on international platforms, including Reddit, YouTube, and other social networks. Traditional models, such as BERT, or commercial solutions, like Perspective API, show significantly lower efficiency in this aspect, as they are primarily configured to work with the English language.

Creating your own case of manually marked comments has become the basis for the high accuracy of our system. This corpus contains real-world examples of texts from various social platforms representing the full range of political positions in their natural context. This approach enables the system to learn to recognise not only direct statements, but also complex forms of expression of position, such as sarcasm, metaphors, specific jargon, and other linguistic features characteristic of the Ukrainian information space.

The proposed system goes far beyond a simple classification, offering a comprehensive analytical toolkit. It is capable of generating TF-IDF models and visual representations of vocabulary (word clouds) for each language separately, conducting detailed analyses of keywords in comments based on the identified political position, and identifying manipulative patterns through the study of frequency dictionaries. This level of analytical depth is fundamentally unavailable in commercial "black boxes" like IBM Watson or MonkeyLearn.

The use of open models from the Hugging Face ecosystem, combined with its proprietary preprocessing and analytics logic, creates a flexible system that can be easily adapted to analyse other conflicts or regional discourses. For example, the system can be configured to work with the Middle East or the Balkan region. The ability to expand dictionaries, topic modules, and analytical frames makes the system scalable and suitable for long-term use.

The proposed approach combines the advantages of modern machine learning (classifiers, clustering) with time-tested methods of natural language processing (lexical analysis, noise filtering, specialised dictionaries). This combination provides reliable results even with limited computing resources. For example, when working with neutral comments, the additional use of vocabulary visualisation and TF-IDF filtering made it possible to identify hidden political connotations that implicitly express the author's position.

Disadvantages and limitations of the developed system:

1. Insufficient amount of training data.
2. Difficulty in interpreting ambiguous statements.
3. Critical dependence on the quality of pre-treatment.
4. Lack of bot identification mechanisms.

5. Real-time processing restrictions.
6. The need for constant updating of dictionaries.

Despite the high quality of manual markup, the size of our case remains relatively limited, which can negatively affect the model's ability to generalise to new data. To fully scale the system, a significant expansion of the corpus is necessary, especially for languages that are currently represented by fewer examples. A large part of the comments on social networks are characterised by ambiguity, irony, or neutrality in wording. Such texts pose challenges for interpretation, even for experienced analysts, let alone automated systems. It is especially true for texts where the political position is expressed through allusions or cultural references. The stage of data preprocessing has a significant impact on the system's final results. Errors in automatic language detection, tokenisation, or removal of noise elements can significantly reduce classification accuracy and lead to incorrect conclusions.

The current version of the system does not distinguish between content created by real users and content generated by bots or automated systems. It can lead to a distortion of analytical conclusions, especially given the active use of bots in information operations. Currently, the system operates with local datasets and lacks the functionality to automatically collect and process new comments from platforms like Reddit in real-time. It limits its use for monitoring current information trends. Dictionaries of terms, emotional markers, propagandistic and manipulative constructions need to be regularly updated in accordance with the evolution of the information field and the emergence of new narratives and formulations.

Despite the active development of NLP tools, most approaches either overlook the political context or require significant adaptation to it. A combination of manual markup, thematic analysis, and manipulation detection distinguishes our project. It makes the task relevant and innovative in the context of information warfare.

4. System Software Analysis

Comments are collected from Reddit, which allows you to cover different points of view regarding the war in Ukraine. For data preparation, CSV files with pre-collected messages were used. Each message contains text, language, source, and other metadata. For accurate analysis, messages are processed in at least three languages: English, Ukrainian, and Russian. At the first stage, the data is cleared of noise, removing links, special characters, repetitions, and unnecessary spaces. Speech detection (langdetect) is also performed.

Based on the sentence-transformers/multilingual-e5-base model, vectorisation is performed for all messages. This model enables you to represent texts in a single multidimensional vector space, regardless of language. It is critically vital for multilingual analysis. The resulting embeddings are used to cluster thematic comment groups, allowing you to identify the most discussed topics (such as weapons, referendums, aid, refugees, economy, etc.).

For English-language messages, the VADER library is utilised to determine the sentiment as positive, negative, or neutral. For Ukrainian and Russian texts, an extension with support for a custom dictionary is provided. Additionally, a hand-compiled dictionary of potentially manipulative vocabulary was compiled. It includes words like "fascists", "Nazis", "liberation", and "ukroregime", which are often used in propaganda narratives. Counting the frequency of such terms enables you to identify comments that may contain potential misinformation.

The classification of the commentary's political position as Pro-UA, Pro-RU or Neutral has been implemented. For this purpose, an approach using poorly labelled data and heuristics was employed, based on keywords and frames inherent to a particular position. An extension with additional training of the BERT or XLM-RoBERTa model on the marked dataset is provided. At the same time, even the basic heuristics showed promising results for identifying extreme positions. Each comment group (language, cluster, stance). A calculation of the number of comments by political positions and tone has also been implemented. To reduce the dimensionality of vector

representations, PCA and t-SNE were employed to visualise the clusters. The purpose of the system is to analyse the political position and sentiment of Reddit users' comments on the war in Ukraine:

1. Automated collection and aggregation of Reddit comments.
2. Detection of the language of texts and pre-processing (cleaning, normalisation).
3. Thematic grouping of comments (clustering).
4. Analysis of the emotional tone of comments.
5. Definition of political position (Pro-UA, Pro-RU, Neutral).
6. Visualisation of results in a format that is convenient for analysis.

We will describe in detail the functional requirements for each module of the information system, which analyses the political position and tone of Reddit users' comments on the war in Ukraine.

Data collection and preparation module:

1. Uploading and preprocessing Reddit comments.
2. Definition of the language of comments (English, Ukrainian, Russian).
3. Cleaning texts from noises, symbols, emojis, and links.
4. Saving data in a structured form (CSV).

Word Processing Module:

1. Tokenisation, lemmatisation of texts.
2. Elimination of grammatical and stylistic errors.
3. Support for custom vocabulary (slang, propaganda templates).

Thematic clustering module:

1. Vectorisation using sentence-transformers.
2. Building thematic clusters (HDBSCAN).
3. Visualisation of clustering results.

Sentiment analysis module:

1. Analysis of the emotional component of comments (Stanza, custom methods).
2. Division into positive, negative, and neutral messages.

Political position classification module:

1. Heuristic definition of stance based on keywords.
2. Additional training of the model on a manual dataset to improve accuracy.
3. Building and testing models (SVM, XGBoost, BERT).

Visualisation module:

1. Building graphs, word clouds, and PCA visualisations.
2. Interface to view classification results and statistics.

Non-functional requirements for the system for analysing the political position and sentiment of Reddit users' comments on the war in Ukraine:

1. Modular system structure for flexible adaptation.
2. Scalability for different languages or data sources.
3. Open source that allows reuse.

4. Use of open source software (Python, Jupyter, Streamlit).

Limitations of the system for analysing the political position and tone of Reddit users' comments on the war in Ukraine:

1. The primary source of data is Reddit (without integrating the APIs of other social networks).
2. Heuristics – the initial method for classifying stance.
3. Partial analysis support for only three languages.

Python 3.11.9 was used to implement the system, specifically the libraries Pandas, NumPy, Scikit-learn, Matplotlib, Seaborn, Stanza, XGBoost, sentence_transformers, tqdm, joblib, UMAP, hdbscan, and langdetect, as well as Matplotlib and Seaborn for visualisation. The environments are Jupyter Notebook (for EDA, clustering, and hypothesis testing) and VS Code (for modular programming, deployment, and integration). The architecture of the system for analysing the political position and sentiment of Reddit users' comments on the war in Ukraine:

1. src/data/ – modules for collecting and saving comments.
2. src/preprocessing/ – cleanup, tokenisation, normalization.
3. src/embedding/ – vectorisation of texts.
4. src/clustering/ – clustering.
5. src/classification/ – stance classification.
6. src/visualisation/ – plotting.
7. notebooks/ – experiments and analyses.

Expected results from the functioning of the system for analysing the political position and sentiment of Reddit users' comments on the war in Ukraine:

1. Structured multilingual Reddit comment data.
2. The language, tone and political position of each message are defined.
3. Thematic clusters have been built.
4. Web interface to visualise results.

The developed information system comprises several main functional subsystems that together provide a full cycle of data work, from collection to visualisation of results.

The data collection subsystem is responsible for obtaining information from Reddit. It provides the download of pre-compiled comments in CSV file format. Each entry contains the text of the message, user ID, language, date, and metadata about the subreddit. In the future, integration with the official Reddit API will enable the extension of functionality and automatic database updates.

The data preprocessing subsystem encompasses cleaning, normalisation, noise removal, and language detection, utilising the langdetect module. It integrates modules for tokenisation, lemmatisation, and cleaning, as well as dictionaries for abbreviations, slang, and propaganda vocabulary. Particular attention is paid to interlingual normalisation for the comparability of analysis results in different language groups.

The modelling subsystem implements multi-stage processing, including clustering of topics using HDBSCAN (with visualisation via PCA/t-SNE), analysis of emotional tonality through Stanza and user dictionaries, and determination of a political position. Classification is carried out both on the basis of heuristic rules and using machine learning (BERT, XGBoost).

The visualisation subsystem provides the construction of graphs, word clouds (WordCloud), clustering and classification results. Additionally, it provides users with a Streamlit web interface to view and filter results by language, topic, political position, and date.

System user roles:

1. Administrator – configures system settings, re-trains models, adds new dictionaries, and monitors the safety and relevance of data. Can perform complete reconfiguration of classification and processing modules.
2. User – interacts through the Streamlit web interface, chooses a language, topic, or position to view analytics. Can compare results by different parameters and export graphs.

Interconnections of components:

1. Data collection modules (data_loader.py) read messages from CSV sources and form a primary database.
2. Preprocessing (clean_text.py, langdetect, preprocessing.py) normalises and cleans up the text.
3. The embedding/clustering modules vectorise the texts and store the coordinates for PCA/t-SNE.
4. The classification/ modules determine the political position and tone.
5. Visualisation modules form graphs, word clouds, and interactive comparisons.
6. The Streamlit interface provides user interaction with the results and allows you to save them as reports.

Information flows:

1. Collection flow – data comes from Reddit or saved CSV files ⟶ data/raw/.
2. Processing flow – data is cleaned, normalised, and distributed by languages ⟶ data/processed/.
3. Clustering flow – embeddings are created, topics are grouped ⟶ data/clusters/.
4. Analysis flow – texts are classified by stance and tonality ⟶ data/results/.
5. Visualisation flow – the results are aggregated, metrics and graphs are generated ⟶ Streamlit.

The system's architecture is implemented on a modular principle, with a clear division into functional components corresponding to the main stages of data processing and analysis, from collection to visualisation of results. Each module can function autonomously, which allows you to scale the system when changing input data, expanding languages or sources, or introducing new analytical approaches. Each module corresponds to a separate processing stage, making it easy to scale the system, add new sources or languages, and deploy individual parts of the project independently. The entire architecture is designed with a focus on reproducibility, transparency, and modularity. Having separate directories for models, dictionaries, results, and raw data makes it easy to debug and retrain individual components. To start the project, install the dependencies listed in requirements.txt and run the Gradio interface using the Python command src/app.py.

The main user interactions with the system (scenarios):

1. View and filter visualisations.
2. Data upload and processing.
3. Running clustering and modelling.
4. Counting manipulative vocabulary.
5. Training or additional training of the model.
6. Updating dictionaries or parameters.

The sequence of the main stages of data processing:

1. Downloading dataset (CSV files) from Reddit (comments).
2. Text cleaning and normalisation.
3. Definition of the message language (langdetect).

4. Lemmatisation and noise removal
5. Vectorisation (embeddings) using the multilingual-e5-base (sentence-transformers) model.
6. Search for manipulative vocabulary.
7. Counting terms from the dictionary.
8. Clustering messages (e.g. HDBSCAN).
9. Classification of political position or tonality,
10. Heuristics/ML model.
11. Combining results.
12. Graphing and word cloud.
13. Output of results and visualisation in the Gradio interface.

It enables you to comprehend the system's logic, from data receipt to result display for the end user. The following diagram describes the structure of our system's program classes. Each class has its own responsibility (Fig. 1):

1. DataLoader – Responsible for reading raw data from CSV.
2. TextPreprocessor – processes text, including cleansing, lemmatisation, and normalisation.
3. Embedder – forms a vector representation of the text.
4. ClusterBuilder – performs thematic clustering.
5. Classifier – determines the political position (pro_Ukraine, pro_russia, neutral).
6. Lexicon Analyser – analyses the presence of manipulative/propagandistic vocabulary.
7. Visualizer – generates output in the form of graphs, tables, and word clouds.
8. GradioInterface is an interactive interface for user interaction.
9. OpinionAnalysisSystem is the main class that coordinates the work of other components.

It allows you to better structure the code logic and relationships between objects.

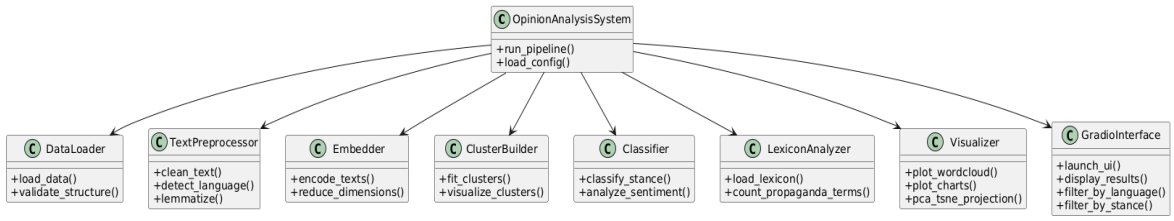

Figure 1: Class Diagram.

The component diagram displays the architectural structure of the system at the component level (Fig. 2):

1. Gradio is a client interface.
2. Core System – implemented in the form of Python modules.
3. The propaganda dictionary is a separate component that is used to detect manipulative messages.
4. CSV source (Reddit Dataset) – input.
5. Data Storage – raw/processed/clusters/results.
6. Visualisation Module – creating graphs, cluster diagrams, and word clouds.

Each component communicates through a specific interface/function, ensuring the scalability and maintainability of the system.

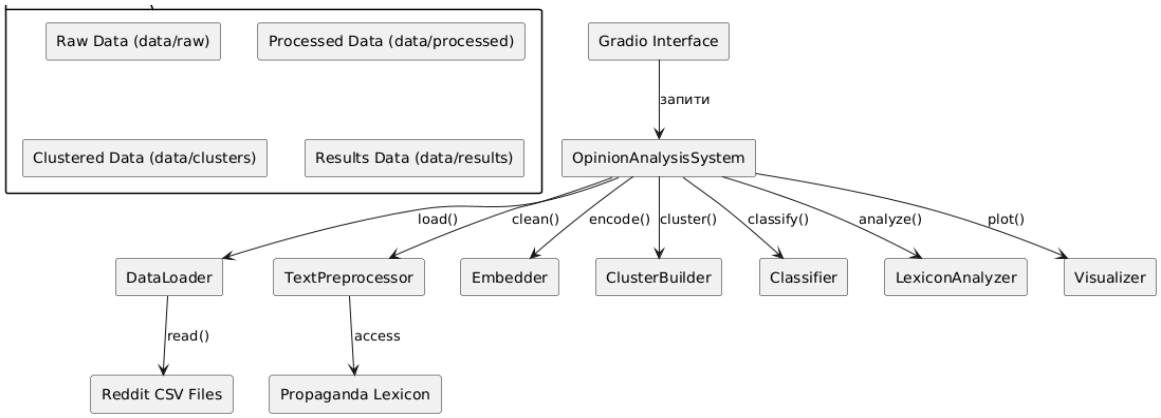

Figure 2: Component Diagram.

A systematic approach to building an information system for analysing public opinion in Reddit social networks has been implemented, with a focus on classifying political positions (stance detection), analysing discussion topics, and identifying manipulative vocabulary. We have gone through a complete cycle:

1. From collecting and preprocessing multilingual comments.
2. To vector representation and thematic clustering.
3. To the classification of political orientation.
4. And building analytical visualisations through the Gradio interface.

The developed system supports multilingual analysis, is tailored to the specific needs of the Ukrainian context, and features a modular architecture that provides flexibility and the potential for further scalability. The results of the work demonstrate that even within a limited dataset, it is possible to build a system capable of effectively analysing political narratives, which is of practical value in the context of media literacy, combating disinformation, and researching the impact of information in wartime.

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

When creating a system for analysing public opinion in social networks, it was necessary to carefully select the tools and methods for each stage of data processing. Particular attention was paid to the specifics of the Ukrainian language and available computing resources.

Python became the primary programming language, and this decision had sound reasoning. According to the TIOBE Index for June 2025, Python ranks first among programming languages, with a score of 25.87%, which is particularly notable in the field of data analysis. When compared to alternatives, Python showed the best balance of characteristics. If R is traditionally strong in statistical analysis, but loses in development speed, and Java demonstrates high performance, but requires significantly more time to write code, then Python combines development speed with sufficient functionality. The main advantage of Python turned out to be its ecosystem. The Pandas library for working with data, scikit-learn for machine learning, Transformers for working with modern language models, spaCy for text processing, and Gradio for creating interfaces have greatly simplified development. The interpreted nature of the language allowed for quick testing of ideas and making changes without a lengthy compilation process. This choice proved especially successful for working with Ukrainian text, as many specialised libraries offer robust support for multilingualism and the ability to fine-tune models for specific tasks.

Reddit was chosen as a source of information for analysing public opinion. It is a platform with active discussions and a substantial amount of Ukrainian-language content, making it an ideal fit for our task. However, there were problems with the official Reddit API. Firstly, there are stringent restrictions on the number of requests – you can only get a limited number of posts in a specific

time. Secondly, a complex authentication procedure that requires registering an application and obtaining special keys. And finally, the data comes in the form of bulky JSON structures that require additional processing. Therefore, we decided to go the other way – to use ready-made CSV files with Reddit data. Such files can be obtained through third-party services or pre-assembled independently. This approach made it possible to focus on the main thing – qualitative data analysis, and not the fight against technical limitations. The CSV files collected all the necessary information: the text of the comment itself, the language of the message, the name of the subreddit, the publication time, and the author's unique identifier. It is pretty enough for a thorough analysis of sentiment. Comparing the two approaches, CSV files emerge as the more straightforward and more accessible option. There are no restrictions on requests, no need to set up complex authentication, and full access to historical data is available. The only drawback is that the data is not updated in real time, but this is not critical for our analysis.

Text processing before analysis is probably the most crucial stage of work. The accuracy of all further conclusions depends on how well we clean and prepare the data. And with Ukrainian texts in social networks, this is especially challenging – there are slang terms, abbreviations, surzhyk with Russian, and various types of jargon. We had to collect a whole arsenal of tools. To determine the language, we used the langdetect library, which recognises Ukrainian well among other languages. With the help of NLTK and ordinary regular expressions, links, user mentions, emojis and other "digital garbage" were removed. However, the most challenging aspect was the morphology of the Ukrainian language. NLTK works well with English, but its capabilities are limited for Ukrainian. spaCy is fast, but also not very friendly with our language. I had to look for a specialised solution. The use of Stanza was considered because it provides a complete morphological treatment of the Ukrainian language. However, due to the complexity of integration, most of the processing is implemented manually or using its own dictionaries and regular expressions.

A particular challenge in analysing texts from social networks is the presence of spelling mistakes, informal word forms, and slang. Therefore, a separate subsystem has been implemented to correct and normalise such cases.

At the planning stage, there was a choice: use large ready-made dictionaries or develop your own approach. Many text analysis systems rely on massive lexical databases – dictionaries with millions of words, corpora of texts, and complex error correction algorithms. But we went the other way. We decided to create a minimalist text cleaning system that does not require external dictionaries. First, large dictionaries occupy a significant amount of space and slow down processes. Secondly, they often fail to take into account the specifics of social networks, such as new slang, abbreviations, and meme vocabulary. Third, it is more critical for our sentiment analysis task to preserve the overall context than to correct every word ideally.

Instead of complex spelling correction algorithms, they focused on the effective removal of "noise" – everything that interferes with the analysis of the content. We have developed a system for step-by-step text cleaning. First, we remove technical junk: URLs that convey nothing about the author's mood; mentions of users of the form @username that are just links; numbers and numbers, if they do not carry an emotional load; and excessive punctuation, which can confuse the analysis. Then we work with the case – bring the entire text to lowercase. It avoids a situation where the words "GOOD" and "OK" are perceived by the system as different. At the same time, we remove unnecessary spaces and emojis – although emojis can carry an emotional load, their analysis requires separate algorithms. Separately, we process stop words – prepositions, conjunctions, and pronouns, which occur in each sentence, but do not affect the mood. So far, we are using the standard English list, but we plan to add a Ukrainian version.

One of the most important steps is to determine the language of the text. In the Ukrainian segment of Reddit, posts in Russian, English, and Polish are often encountered. For our analysis of public opinion in Ukraine, such content may be irrelevant or even noise in the results. Therefore, at the very beginning of the processing, each text is run through langdetect. This library accurately

determines the language, even for short messages. If the text is not in Ukrainian, we immediately discard it, without wasting time on further processing. It significantly speeds up the system.

When you process thousands of messages, every millisecond counts. Therefore, we compile all regular expressions in advance using the re.compile function. It means that Python "understands" the search pattern once and then applies it to each text. The sequence of processing also matters. First, a quick language check – if the text does not fit, we immediately move on to the next one. Then the fastest operations are to remove URLs and mentions. And only at the end more complex transformations. The result of this approach is that the system can process a case of tens of thousands of messages in a matter of minutes on a regular computer. At the same time, you do not need to download and store large dictionaries in memory.

The simplicity of the architecture makes it easy to understand what is happening at every step. If something doesn't work as expected, it's easy to identify and resolve the issue. There are no "black boxes" in the form of complex linguistic algorithms. The speed of work remains high even with an increase in the amount of data. The system does not rely on external resources – dictionaries that require constant updates. Flexibility for expansion – if you need to add new types of processing in the future, this can be done in stages, without rewriting the entire system from scratch. Of course, this approach has its limitations. We do not correct typographical errors, do not recognise complex grammatical constructions, and do not analyse syntax. However, for the task of analysing moods, this is reasonably sufficient – the main thing is to preserve the general meaning and emotional tone of the text.

The task is how to convert words into numbers that can be processed by machine learning. After all, the computer does not understand what "good" or "bad" is – it works only with numbers. Therefore, it is necessary to find a way to represent each text in the form of a vector of numbers so that texts similar in content have identical numerical representations. We tested two radically different approaches to this problem – the classical statistical method and modern neural networks. Each has its own advantages and disadvantages.

We started with TF-IDF, which is an abbreviation for Term Frequency-Inverse Document Frequency. It may not sound very easy, but the logic is actually elementary. Imagine that you are analysing political commentary and want to understand which words are most important to determine the author's position. TF-IDF works according to the principle: if a word is often found in a particular document but rarely in the entire collection, then it is likely vital for that document. For example, the word "president" can occur in many political texts, so its weight will be less. However, some specific words that are characteristic of a particular political position will receive higher weight. The advantages of TF-IDF are obvious: speed, ease of implementation, and clarity of results. You can easily see which words the system considers most important for each class. No powerful graphics cards or special computing resources are required. But there are also serious drawbacks. TF-IDF does not understand the context at all. For him, the phrases "Ukraine defeated Russia" and "Russia defeated Ukraine" are very similar, as they contain the exact keywords. He does not take into account the order of words, does not understand synonyms, and fails to recognise semantic connections. When we tested TF-IDF on our data, the results were, to say the least, not impressive. He performed particularly poorly with short comments, where context is critical, given the limited number of words to analyse. On social networks, most messages are brief.

Therefore, they switched to modern methods – contextual embeddings. It is a result of the revolution in natural language processing that has occurred in recent years, thanks to the development of neural networks of the Transformer type. The primary concept of contextual embedding is to encode not individual words, but entire sentences, considering the surrounding context. A neural network trained on massive corpora of texts "understands" semantics – it knows that "good" and "excellent" are similar in meaning, what "not bad" is actually positive, that "Ukraine won" and "Russia lost" convey similar information. For our project, we used SentenceTransformer – a family of models specially configured to create vector representations of entire sentences. We chose a multilingual model that supports the Ukrainian language. It is critically important because the most popular models are primarily trained on English text. Each sentence is fed to the input of

a neural network that consists of hundreds of millions of parameters. At the output, we get a vector of several hundred numbers that encode the meaning of the sentence. At the same time, sentences with similar meaning will have similar vectors – they can be compared using cosine distance or other metrics.

The difference between TF-IDF and contextual embedding is especially noticeable in examples from Ukrainian politics. Let's take two comments: "Zelensky is leading the country well in difficult times", "The president is not doing his job well." TF-IDF sees only the words: "Zelensky", "good", "leads", "country" in the first and "president", "bad", "copes" in the second. For him, these are two different texts with minimal word intersection. Contextual embedding recognises that "Zelensky" and "president" in the Ukrainian context often refer to the same person. They hear the opposite tone: "leads well" versus "does not do well". Such texts will receive vectors that will be far apart in the vector space.

Working with the Ukrainian language added its own difficulties. Most of the best models are trained on English text. Although multilingual models exist, their quality for Ukrainian may be worse than for more "popular" languages. I had to experiment with different models and test their behaviour on the Ukrainian text. It turned out that some models confuse Ukrainian with Russian or Polish, especially when there are many borrowings in the text. But when a suitable model was found, the result exceeded expectations. The system began to understand subtle differences in political vocabulary, recognise sarcasm, and consider the context of Ukrainian realities.

Ultimately, it was decided not to abandon the TF-IDF entirely. Although contextual embedding produces better quality, it is also much slower. For the initial selection and rapid analysis of large datasets, TF-IDF remains useful. We created a hybrid system: first, TF-IDF helps to filter out clearly irrelevant texts and make a rough classification. Contextual embeddings then provide subtle analysis for the most critical messages. This approach enabled the achievement of the best of both worlds: the speed of classical methods and the accuracy of modern neural networks. Additionally, the system has become more stable; if complex models fail for any reason, you can always revert to reliable yet straightforward methods.

Contextual embedding has proven to be useful not only for classifying political positions but also for other applications. We also used them for clustering, which involves finding hidden groups of users with similar views. When you have high-quality vector representations of texts, a variety of machine learning algorithms can be applied: from simple classification to complex analysis of social networks. Vector space allows not only to classify texts, but also to search for similar ones, identify anomalies, and track changes in public opinion over time. It opens up vast opportunities for further development of the system.

Several models from the scikit-learn library were used to classify the political position. The main goal is to find a balance between learning speed, accuracy, and interpretation.

1. The Logistic Regression model has become the basis of our project. It is easy to implement, provides good accuracy on small amounts of data, and is easy to interpret. It has been tested with both TF-IDF and SentenceTransformer vectors.
2. K-Nearest Neighbours (KNN) — this approach allowed you to work better with data clusters. Its advantage is intuitiveness and the ability to process new points by similarity to known ones. It was used mainly with TF-IDF vectors.
3. The XGBClassifier model (an ensemble method — gradient boosting) yielded the highest accuracy, but required more time to learn. Due to its good generalisation, XGBoost was effective on contextual vectors.

We evaluated the quality of the classification according to the following metrics:

1. Accuracy – the overall accuracy of the classification.
2. Precision and Recall – for each class (pro-Ukraine, pro-Russia, neutral).
3. F1-score is the harmonic mean of Precision and Recall.

Technical metrics:

1. Confusion Matrix – to identify common classification errors.
2. Cosine Similarity – to check the quality of vectorisation between similar messages.
3. Loss curve graph – to monitor learning and avoid overlearning.
4. Execution time – to evaluate the effectiveness of each approach (TF-IDF vs transformers).

These metrics enabled a comprehensive assessment of the effectiveness of both classical models and modern transformer approaches.

To visualise the results of the classification, the following libraries were used:

1. Matplotlib – for plotting basic graphs, including PCA visualisation of clusters.
2. Seaborn – for building Confusion Matrix heatmaps and distribution graphs.
3. Pandas – for building tables with analysis of results.

The main objective was to depict the spatial structure of clusters and the distribution of classes. The libraries were chosen because of their flexibility and ease of integration with other parts of the pipeline. Streamlit was not used in this project, as the focus was on offline analytics, clustering, and graphical interpretation of results in Jupyter Notebook.

The project was developed in the Jupyter Notebook environment, utilising Visual Studio Code as the editor. The combination of these tools made it possible to effectively:

1. Develop, run and test code in stages.
2. Visualise the results of the classification.
3. Save the model's logic and conclusions in a format convenient for presentation.

VS Code provided advanced autocompletion, debugging, project navigation, and GitHub synchronisation. Jupyter has been the primary medium for experimentation and model testing. It offered a transparent project structure, convenient scheduling, and a step-by-step recording of results.

Various approaches to vectorisation (TF-IDF and transformers), classification (logistic regression, XGBoost, KNN), and evaluation of results (F1 score, accuracy, confusion matrix) were analysed. Selected Python platform tools, such as scikit-learn, transformers, matplotlib, and pandas, made it possible to achieve high classification accuracy while maintaining the simplicity of implementation and the ability to scale the system. The project demonstrated the potential of automated analysis of public opinion in social networks, opening up opportunities for further use of models in media monitoring, political analytics and digital sociology.

6. Software Development

The development of the Reddit comment sentiment analysis system required the integration of natural language processing (NLP) methods, machine learning algorithms, and specialised approaches to handling content that contains political and emotionally charged language, as well as potential manifestations of manipulative rhetoric. A notable feature of Reddit comments is their linguistic variability, encompassing multiple languages (English, Ukrainian, Russian), diverse communication styles, the presence of spelling errors, abbreviations, Internet slang, and memetic elements. As a result, the system had to be adapted to accommodate multilingual content, capable of revealing both explicit and implicit user positions regarding the war in Ukraine. The architecture of the project is built on the principles of modularity, where each stage of processing is implemented in the form of separate modules:

1. Module for downloading and filtering Reddit comments (CSV, JSON).
2. Text preprocessing module (cleaning, language detection, normalisation).

3. Text vectorisation module (SentenceTransformer multilingual-e5-base).
4. Manual markup and machine learning module (Logistic Regression).
5. Clustering and visualisation module (KMeans, PCA).
6. User Experience Module (Gradio).

In the pre-processing phase, the removal of URLs, user mentions, punctuation marks, and numbers, as well as the conversion of lowercase text, has been implemented. Comments were filtered by language, allowing you to focus on English-language messages for initial markup and model training.

Vectorisation is implemented based on the multilingual-e5-base contextual model, which enables the generation of a deep vector representation for each comment, taking into account its context, intonation, and semantic load. It significantly improves the classification quality compared to TF-IDF.

The classification model was trained on a manually marked subset dataset with the positions: pro-Ukraine, pro-Russia, and neutral. Logistic Regression was chosen as the classifier due to its stability, ease of interpretation and good consistency with linearly represented embeddings.

Visualisation of results is carried out through PCA dimensionality reduction and clustering, which allows groups to be tracked by key and position. To demonstrate the analysis, an interface has been implemented on Gradio, which allows you to enter a comment, receive predictions, and view examples of classified comments.

Thus, the system encompasses the entire cycle of social content processing, from collection and purification to vectorisation, classification, and interactive demonstration of the results of public opinion analysis.

The software is implemented as a modular system with a clear division of responsibilities between components. The architecture is built on the principle of a multi-level model, where each layer performs separate data processing functions:

1. Level 1 — downloading and filtering data (English-language Reddit comments).
2. Level 2 — preprocessing of text (cleanup, lowercase, deletion of URLs, characters, etc.).
3. Level 3 — vectorisation (SentenceTransformer multilingual-e5-base).
4. Level 4 — training the classification model and saving the results.
5. Level 5 — clustering and visualisation (KMeans, PCA).
6. Level 6 — user interface (Gradio application for demonstration).

The project has the following structure:

1. data/ – saved files with Reddit comments and classification results, in particular, raw/ (initial CSV files with data) and processed/ (cleaned and marked up data);
2. models/ – stored classification models and vectorizers;
3. embeddings/ –.npy embedding files;
4. notebooks/ – Jupyter notebooks for EDA, clustering, model training;
5. src/ – main code: preprocessing.py – text cleaning functions; training.py – training code for the logistic regression model; vectorization.py – work with multilingual-e5; clustering.py – clustering, PCA imaging; interface.py is the implementation of the Gradio application.

The entire system is designed to be expandable for new platforms (such as Twitter and YouTube) or adaptable to other languages and categories.

Creating a high-quality dataset for classifying the sentiment of Reddit comments about the war in Ukraine required careful planning of each stage, from data collection to final markup. The focus was on the relevance of the content, the purity of the language, and the accuracy of the annotation.

The source of the data was popular subreddits that covered the Ukrainian-Russian war, international support, political leaders, and the social consequences of the conflict. The collection

was conducted via API or the export of available datasets, followed by saving the data in CSV format. The collected comments underwent multi-stage filtering:

1. Removal of non-informative fields and empty comments.
2. Detection of the language of the comment (English, Ukrainian, Russian).
3. Automatic translation of Ukrainian and Russian comments into English (for unification of vectorisation).
4. Cleaning the text of HTML artefacts, URLs, symbols, and mentions.

The main feature of the markup was the integration of the cluster approach. Initially, each comment was vectorised using the multilingual-e5-base transformer model, which creates semantically rich representations of sentences in a multilingual space. Next, K-Means clustering was used. After clustering, the most representative examples of each cluster were manually analysed to identify the prevailing political position. Based on this, each cluster was assigned one of the following labels: +1 (pro-Ukraine), 0 (neutral), or −1 (pro-Russia). Thus, it was automatically distributed to all comments in the corresponding cluster. It made it possible to achieve a compromise between speed and quality, as well as to reduce the influence of the human factor when marking up large amounts of data. Manual verification of clusters ensured the correctness of the results and allowed for the identification of cases of semantic noise or ambiguity.

The text preprocessing module is a critical component of the entire system, since the quality of further analysis directly depends on the purity of the input data. Reddit comments contain a significant amount of "noise": HTML tags from formatting, URL links, special characters, redundant spaces, and various artefacts that can negatively impact the accuracy of NLP models. The advantages of the developed approach include the preservation of semantics, processing of multilingual content, and optimisation of performance. Unlike aggressive cleanup, which can remove important information, our approach preserves key elements of the text by replacing the URL with a [URL] marker, allowing the model to understand that a link was present in the text. Unicode normalisation ensures the correct processing of texts from various language systems, including Cyrillic, Latin, and other alphabets. Regular expression compilation and batch processing significantly enhance the speed of processing large datasets.

```
import re
def clean_text(text):
    text = re.sub(r'http\S+', '', text)
    text = re.sub(r'<.*?>', '', text)
    text = re.sub(r'[^\w\s]', '', text)
    return text.lower().strip()
```

One of the biggest challenges when analysing social media content is multilingualism. Reddit communities comprise users from all over the world, who naturally write in their native languages. Direct analysis of such multilingual content results in data fragmentation and a decline in classification quality. The key advantages of the translation system are intelligent caching, contextual language definition, error handling and fallback mechanisms, and optimisation of API calls. An LRU cache with a capacity of 10,000 elements prevents repeated translations of identical texts, thereby significantly increasing processing speed and reducing the load on the translator's API. The system utilises not only automatic language detection but also heuristic rules to distinguish between similar languages (e.g., Ukrainian and Russian), which is especially important in the context of political discussions. In the event of an unsuccessful translation, the system returns the original text, ensuring the stability of work even in the event of technical problems with translation services. The system incorporates latency and batch processing to comply with rate-limiting limits for external services.

```
from langdetect import detect
from deep_translator import GoogleTranslator
def translate_comment(comment):
    lang = detect(comment)
    if lang != 'en':
        return GoogleTranslator(source=lang, target='en').translate(comment)
    return comment
```

Text vectorisation is a key step in converting unstructured text content into numerical representations that machine learning algorithms can efficiently process. Unlike traditional methods such as TF-IDF or Bag-of-Words, modern transformer models create multidimensional vector representations that capture deep semantic connections between words and phrases. The intfloat/multilingual-e5-base model was chosen for vectorisation due to its multilingual support, optimal balance of quality and speed, and contextual understanding. The model is trained on texts from 100+ languages, making it ideal for processing multilingual Reddit content. 768-dimensional vectors provide sufficient expressiveness for complex semantic problems while remaining computationally efficient. Unlike static word embeddings, the model considers the context of each word within a sentence. Advantages of the developed vectorisation system include GPU acceleration, memory optimisation, caching of results, and flexibility in analysis – automatic detection and use of CUDA to speed up computing, with fallback to CPU for non-GPU systems. The use of FP16 on the GPU and batch processing allows you to efficiently process large datasets even on limited resources. The system of saving and loading vectors eliminates the need for repeated calculations on already processed texts. Additional methods for calculating similarity, searching for similar texts, and reducing dimensionality expand the analytical capabilities of the system.

```
from sentence_transformers import SentenceTransformer
model = SentenceTransformer("intfloat/multilingual-e5-base")
embeddings = model.encode(texts, show_progress_bar=True)
```

Classifying political sentiments in social networks is a much more difficult task compared to traditional sentiment analysis. Political commentary often contains:

1. Complex irony and sarcasm: "Thank you, Putin, for 'peace' in Ukraine".
2. Cultural references, i.e. references to historical events that require contextual understanding.
3. Euphemisms and code words: "Special military operation" instead of "war".
4. Multi-layered meanings, in particular, comments, which can have different interpretations depending on the context.

The key advantages of the classification system are: an ensemble approach, intelligent balancing, complex validation, and analysis of interpretation. Combining different algorithms (logistic regression, Random Forest, SVM) allows you to compensate for the weaknesses of individual models and increase overall accuracy. The system utilises SMOTE to generate synthetic examples of minority classes and undersampling to reduce the prevalence of majoritarian courses, thereby ensuring an optimal balance. Cross-validation, along with various metrics (F1-macro, Matthews correlation coefficient, Cohen's kappa), provides an objective assessment of the model's quality. The Feature Importance Analysis System enables you to understand which words and phrases have the most significant impact on classification, which is crucial for comprehending political sentiment.

```
vfrom sklearn.linear_model import LogisticRegression
clf = LogisticRegression(max_iter=1000, class_weight='balanced')
clf.fit(X_train, y_train)
```

Cluster analysis presents a unique opportunity to look beyond a simple three-part classification (pro-Ukrainian, pro-Russian, neutral) and identify more nuanced semantic groups in political discussions. Imagine a vast space of thoughts, where each comment is a point with coordinates that reflect its semantic meaning. Clustering allows you to find natural "clusters" of similar thoughts that can reveal hidden trends that are not noticeable in a superficial analysis. For example, among the "neutral" comments, individual clusters can stand out: one contains truly balanced opinions on a peaceful settlement, another – veiled scepticism about the Ukrainian position, and the third – disappointment with the duration of the conflict. These nuances are critical to understanding the actual dynamics of public opinion.

Working with 768-dimensional vectors presents unique challenges. Imagine that each comment is represented by a point in space with 768 axes – this is absolutely impossible to visualise with the human eye. It is where the "dimensionality curse" arises: in high-dimensional spaces, traditional distance measurement methods become less reliable, and visualisation becomes impossible. Principal component analysis (PCA) solves this problem elegantly: it finds the most critical "directions" in the data – those axes along which the comments differ most from each other. The first two principal components typically capture the most significant differences between commentaries, enabling a two-dimensional "map" of political sentiment.

Clustering reveals psychological patterns in political discussions. Unexpected groups are often found: "passive supporters" (support one side without aggressive rhetoric), "radical critics" (sharply condemn opponents), "analysts" (attempt to understand the complexity of the situation), and "emotional commentators" (react to specific news). Exciting are the "transitional" clusters – comments that are on the border between the main categories. These texts often reflect the internal contradictions of the authors, their doubts or the evolution of views. Analysis of such clusters can reveal how people adjust their positions in response to new events. The creation of two-dimensional maps of political sentiments is of great practical importance. They enable the detection of polarisation, tracking the evolution of thoughts, and identifying influential topics. If the clusters are located far from each other, this indicates a high polarisation of opinions. If they overlap, it means greater tolerance for different views. By comparing maps for various periods, you can observe how moods shift in response to events. For example, after significant hostilities, clusters may become more polarised. The analysis of cluster centres shows which themes or arguments are most characteristic of each group. It helps to see how semantically different the comment groups are.

```
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
pca = PCA(n_components=2)
X_pca = pca.fit_transform(embeddings)
kmeans = KMeans(n_clusters=3)
labels = kmeans.fit_predict(X_pca)
```

Visualising political data is not just about creating visually appealing graphs. It is the art of telling a story that is hidden in numbers. Each visualisation should answer a specific question: "Have moods changed over the past month?" "What words are most often used by supporters of different parties?" "How polarised is society?" A particular challenge lies in striking a balance between scientific accuracy and accessibility to a broad audience. Researchers need detailed statistical graphs, journalists need vivid infographics, and politicians need simple charts for quick decision-making.

People perceive political visualisations through the prism of their own beliefs. It creates unique challenges, including cognitive biases, the emotional impact of colours, and information overload. A person who supports one side may interpret the same diagram differently from a supporter of the opposite view. Therefore, visualisations should be as objective and understandable as possible. The use of red and blue colours in political visualisations is not neutral – it evokes associations with different political forces. The system utilises neutral colours to minimise subconscious influences. Political data is highly multidimensional, but the human brain can efficiently process a limited amount of information at a time. Effective visualisation shows precisely what you need to know, nothing more and nothing less.

Innovative approaches to mood visualisation: word clouds as psychological portraits, heat maps of manipulative vocabulary, and dynamic visualisations of changes over time. Traditional word clouds show the frequency of word use, but our system creates "psychological portraits" of each group. Word size reflects not only frequency, but also emotional weight and uniqueness for a particular group. The system generates unique heat maps that display the frequency with which different groups employ emotionally charged words. It helps to identify propaganda techniques and manipulative techniques. Static graphs only show a snapshot of the situation. The system generates animated visualisations that display the evolution of sentiment over time, identifying trends and reactions to specific events.

Visualising political data carries a huge responsibility. Misrepresented information can affect public sentiment, political decisions, and even international relations. The system ensures that visualisations do not reinforce negative stereotypes about any groups or nations. All graphs are created with the aim of presenting different perspectives fairly, without bias in favour of either side. Each visualisation is accompanied by an explanation of its context and the limitations of interpretation. These visualisations help you understand the distribution of sentiments and key themes in the comments.

```
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
def plot_sentiment_distribution(labels):
    sns.countplot(x=labels)
    plt.title("Sentiment Distribution")
    plt.show()
def plot_wordcloud(texts):
    text = ' '.join(texts)
    wc = WordCloud(width=800, height=400).generate(text)
    plt.imshow(wc)
    plt.axis("off")
    plt.title("WordCloud")
    plt.show()
```

Creating an intuitive interface for analysing political sentiments is a matter of democratising knowledge and information. Powerful machine learning and NLP algorithms have traditionally only been available to professionals with technical education. Our system breaks down this barrier by enabling journalists, researchers, activists, and ordinary citizens to conduct their own research on public opinion.

Imagine a journalist who wants to understand the public's reaction to a new political statement. Instead of spending weeks learning programming and machine learning, it can simply upload comments to the system and get a detailed analysis in minutes. Trust in an AI system does not arise instantly. Users, especially in the context of political analysis, are naturally sceptical of "black boxes" that give conclusions without explanation. Therefore, the interface is designed with the principle of maximum transparency, providing explanations of each step, visualising confidence, and allowing for verification. The system displays the entire analysis process to the user, from text

cleaning to final classification. It enables you to understand how the system arrived at its conclusions. A confidence indicator accompanies each forecast. If the system is uncertain about the classification of a particular comment, it reports it honestly. Users can view examples of comments from each category and evaluate the correctness of the classification for themselves.

The system is designed taking into account the needs of different categories of users:

1. Scientists and researchers require detailed statistics, the ability to export data, and the flexibility to configure analysis parameters. Advanced features and APIs are available for them to integrate with their own research.
2. Journalists appreciate speed and clarity. Quick visualisations and ready-made infographics are optimised for use in publications.
3. Public activists want to understand the mood of their communities. The system provides in-depth but straightforward analytical tools for monitoring public reactions.
4. Politicians and government officials require prompt and accurate assessments of public opinion to inform their decisions. Summary reports and trends are available for them.

Ethical principles of interface development: avoidance of manipulation, privacy protection, openness and transparency. The interface is designed to avoid imposing specific conclusions on users. All results are presented neutrally, with an explanation of limitations and possible alternative interpretations. The system does not store users' personal data and does not track their activity. All analyses are performed locally or with minimal data retention. Algorithms and analysis methods are described openly, allowing users to understand the system's principles and its limitations. The interface is designed to be user-friendly, even for those without technical experience, making it convenient for real-time demonstrations and analysis.

```
import gradio as gr
def analyze_sentiment(comment):
    # preprocessing → translation → vectorization → prediction
    return prediction
gr. Interface(fn=analyze_sentiment, inputs="text", outputs="label").launch()
```

Modern social networks generate colossal amounts of content. On Reddit alone, millions of comments are published every day, and even a small fraction of them related to a specific topic can be tens of thousands of texts. Processing such large amounts of data poses unique technical challenges, including memory issues, processing times, and network limitations. The 768-dimensional vectors for each comment quickly fill up the RAM. For example, storing 100,000 comments requires approximately 300 MB to store vectors, not counting the original text and intermediate calculations. Vectorisation of text using transformer models is a computationally expensive operation. On a conventional processor, processing 10,000 comments can take several hours. Translating comments through external APIs can create delays and impose restrictions on the number of requests per minute.

The system uses a multi-level approach to batch processing. At the first level, comments are grouped by language to optimise translation. At the second level, optimally sized batches are created for vectorisation, taking into account the available memory and computing resources. This approach enables you to process large datasets in parts, without overloading the system, while maintaining efficiency through vectorised operations. Traditional caching stores results for identical inputs. Our system goes further – it uses semantic caching. If the new comment is very similar to the one already processed (based on the cosine similarity of the vectors), the system can use the cached result instead of reprocessing. It is especially effective for comments that are slight variations of each other – a typical situation on social media, where people often rephrase similar thoughts. The system automatically adapts to the available computing resources. On powerful

servers with GPUs, it uses parallel processing and acceleration. On regular computers, it optimises the size of the batches and uses more conservative settings to avoid memory overflow.

The system's modular architecture enables easy distribution of the load across multiple servers. Different modules (translation, vectorisation, and classification) can work on separate machines, communicating via an API. It is vital for real-time monitoring, where it is necessary to handle a constant stream of new comments without delay. The system is designed to account for possible failures. Each processing stage can be restarted from the last successful point. Intermediate results are saved to disk, which allows you to resume work after unforeseen outages. Particular attention is paid to error handling in external services (translation API, model loading). The system has fallback mechanisms for all critical components.

The development of an effective system for analysing political sentiments necessitated in-depth empirical research. We conducted a series of experiments to optimise each component of the system and validate its performance in real-world conditions. The first stage involved comparing different approaches to text vectorisation. We tested traditional methods, Word2Vec and GloVe, as well as Transformer models. The n-gram TF-IDF showed basic efficacy but failed to capture complex semantic relationships in political commentary. The handling of sarcasm and irony was especially problematic. The Word2Vec and GloVe methods performed better with semantic relationships, but had limitations in handling multilingual content and contextual understanding. BERT, RoBERTa, and SentenceTransformers performed significantly better, particularly in understanding context and handling informal social media language. An interesting finding was that multilingual models (such as multilingual-e5-base) performed better, even on English-language texts, compared to monolingual models. It is due to their greater resistance to noise and language variability.

One of the most controversial aspects of our approach is the automatic translation of all comments into English. Critics argued that translation could distort semantics and reduce classification accuracy. To test this hypothesis, we conducted a controlled experiment with 5,000 comments in Ukrainian, Russian, German, and French, which were independently labelled by native speakers. The results were unexpected: the accuracy of classifying the translated texts was only 3–5% lower than that of the original texts, but the overall efficiency of the system increased significantly due to the ability to use a single model for all languages. Moreover, some subtle semantic shades that were lost during translation were compensated for by a greater consistency of classification and the ability to identify interlingual patterns in political rhetoric.

We tested a wide range of machine learning algorithms. Logistic regression showed the best balance between accuracy and interpretation. The coefficients of the model allow you to understand which words have the most significant influence on the classification. Gradient Boosting (XGBoost, LightGBM) exhibited higher accuracy on the test sample but was prone to overtraining and less interpretable. Deep models demonstrated the highest accuracy, but required significantly more computational resources and were largely uninterpretable. Ensemble methods: Combining several algorithms (Ensemble methods) yielded the best result – high accuracy with an acceptable level of interpretation.

Creating a high-quality dataset for training was one of the most challenging tasks. We collected over 50,000 comments from various political subreddits, including r/worldnews, r/europe, r/UkrainianConflict, and others. The marking process included several stages:

1. Automatic pre-filtering to remove spam and irrelevant comments.
2. Markup by a team of 10 annotators, including speakers of different languages.
3. Conflict resolution through voting and discussion.
4. Validation by a separate group of experts in political science.

The inter-notation coherence (Cohen's kappa) was 0.72, which is considered a good result for such a subjective problem as the classification of political sentiments.

It was crucial to test how the system works with comments from different time periods. Political sentiment can change rapidly, and a model trained on data from a single period may not perform well on more recent data. We tested the system on comments from different stages of the conflict:

1. Before the start of the full-scale invasion (2021-2022).
2. The first months of the war (March-May 2022).
3. Later stages (summer-autumn 2022).

A detailed analysis of the errors revealed several characteristic patterns:

1. Sarcasm and irony: "Thank you, Putin, for peace in Europe" – such comments were often classified as pro-Russian due to their superficial content, ignoring the ironic context.
2. Cultural references, i.e. references to historical events or memes, were understandable to native speakers but were lost during automatic processing.
3. Contextual dependence, in particular, refers to comments that require knowledge of previous messages in the thread for proper understanding.
4. Ambivalent statements: "Both sides have the right to exist, but..." – such comments balance between different positions.

To solve the identified problems, we have developed several strategies:

1. The sarcasm detector is an additional classifier for detecting ironic comments with subsequent inversion of their classification.
2. Contextual analysis – taking into account previous comments in the thread for a better understanding of the context.
3. Ensemble methods – combining several models to reduce the impact of specific errors of each.

Each comment in the dataset has the following features:

1. Primary text.
2. Language.
3. Translation into English (if required).
4. Cleaned text.
5. Vector representation (768-d).
6. Class label (pro-Ukraine, pro-Russia, neutral).

To implement the model architecture and represent features, logistic regression with one-hot class vectors and SentenceTransformer vectors (768 dimensions) is used. For visualisations, PCA is used up to 2 dimensions – clustering via KMeans.

Models and data are stored in the models/ (joblib), embeddings/ (.npy) and data/ (.csv) directories. Reuse is done without the need for retraining. The system is deployed via interface.py and gradio.launch().

The developed system for analysing Reddit comments on the war in Ukraine integrates multilingual processing, translation, vectorisation, classification, clustering, and an interface in Gradio. It is adapted to the informal style of social networks, includes visualisations and allows you to effectively explore political sentiments online. Examples of classified comments:

1. Pro-Ukraine: "Glory to Ukraine! They are fighting for democracy and freedom."; "Ukrainians have shown more courage than the whole of NATO combined.".
2. Neutral: "I hope there will be peace soon. This conflict has affected everyone."; "Both sides have lost too much already.".

3. Pro-Russia: "Russia had to respond to NATO's provocation."; "Crimea was always part of Russia, just correcting a mistake.".

These examples are automatically categorised using logistic regression trained on manually labelled data from Reddit.

7. Implementation of a control example

The purpose of the control case is to demonstrate the effectiveness of the automated analysis system of Reddit comments on the war in Ukraine. The system allows (Fig. 3–10):

1. Classify the political position of the commentary.
2. Identify potentially manipulative vocabulary.
3. Translate non-English comments.
4. Display the result in a user-friendly interface with visualisations and explanations.

A representative sample of 7,000 Reddit comments, collected from communities related to the war in Ukraine, geopolitics, NATO, Russia, and the Ukrainian army, was used to conduct the control case. The comments spanned several languages (English, Ukrainian, and Russian) and various styles of discourse (official, sarcastic, memetic, and aggressive). Before classification, the following was performed:

1. Automatic language detection.
2. Translating non-English comments into English (using the Google Translate API).
3. Pre-processing of the text (cleaning, normalisation, lemmatisation).
4. Formation of a vector representation of comments via SentenceTransformer (multilingual-e5-base).

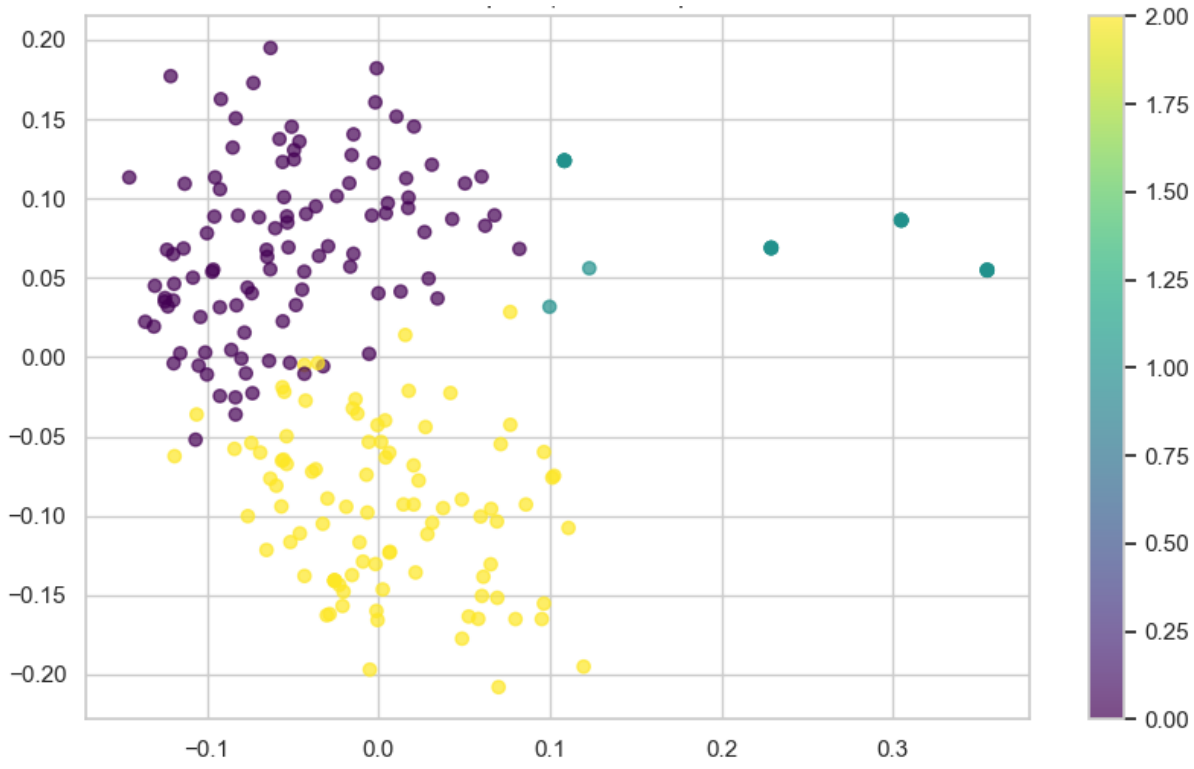

Figure 3: Comment clustering.

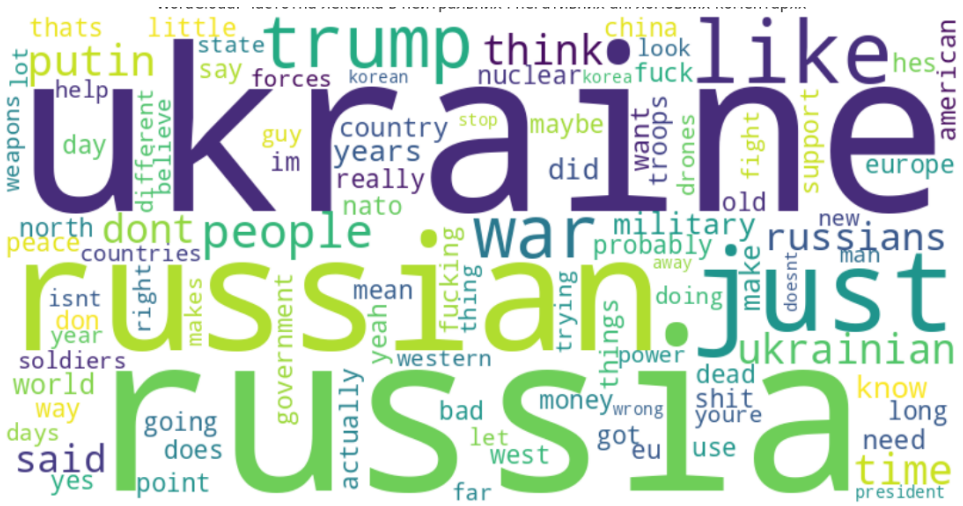

Figure 4: Frequency Vocabulary in Neutral and Negative English-Language Comments.

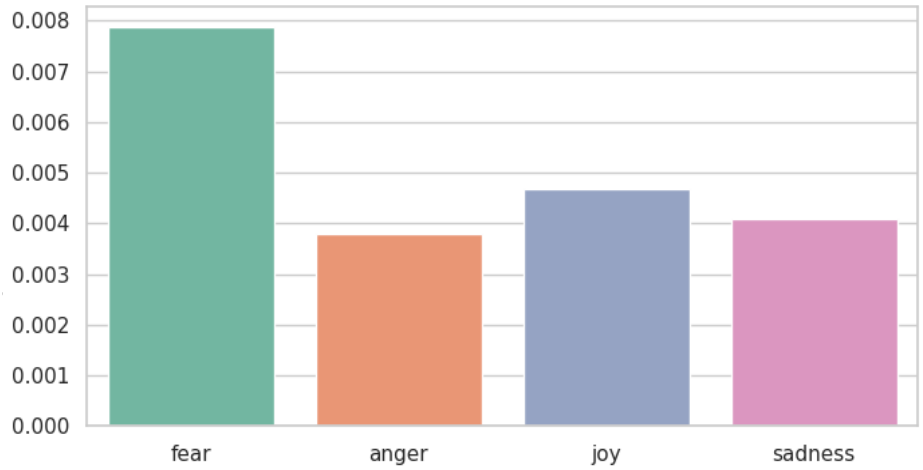

Figure 5: Average number of emotional words per English comment.

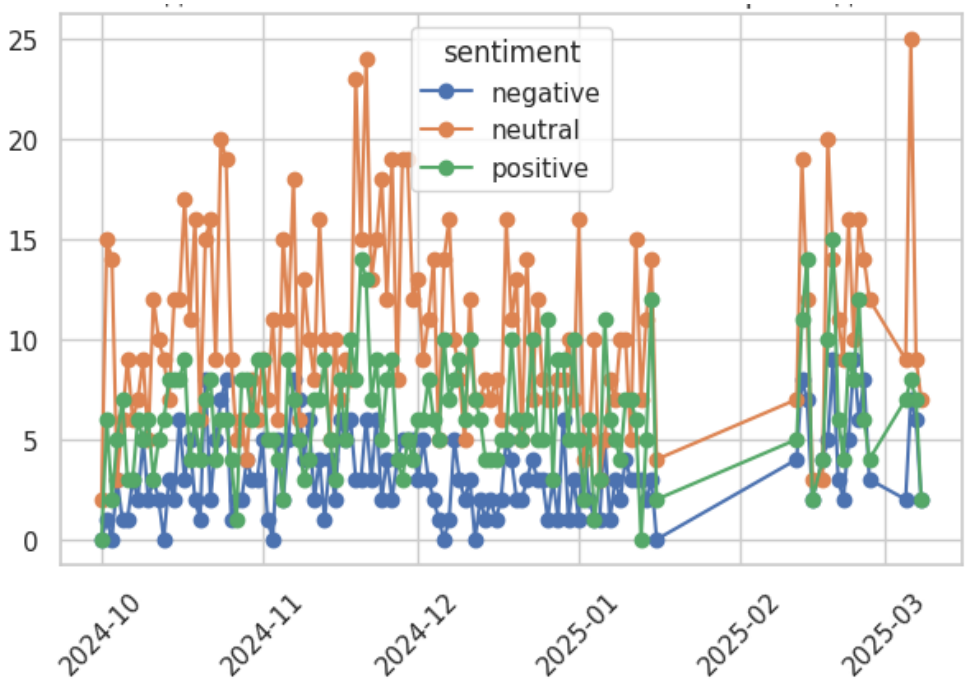

Figure 6: Distribution of keys of English-language commentaries depending on time (by dates).

Figure 7: Most Frequent Words in Positive Comments (TF-IDF).

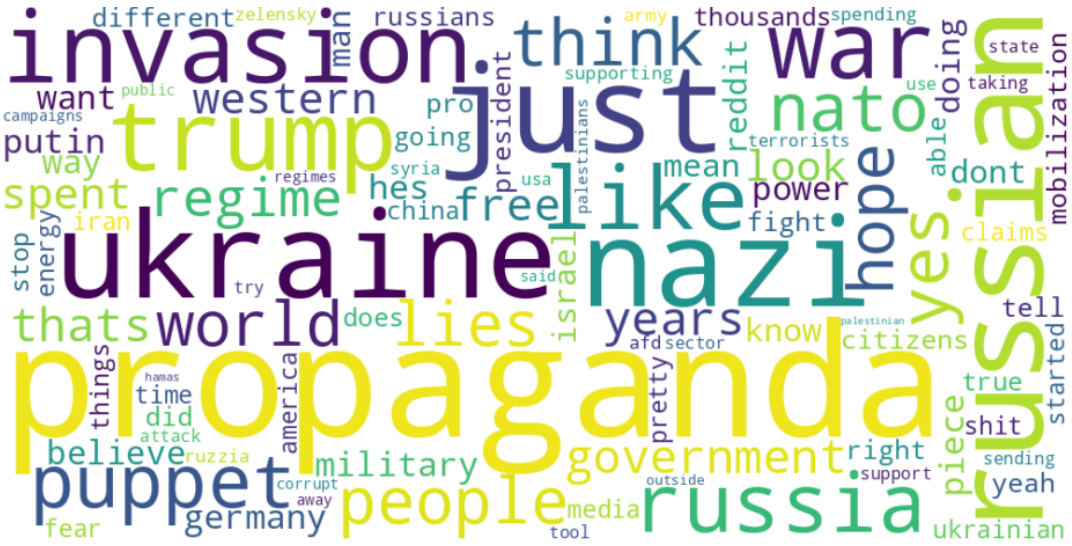

Figure 8: Most Common Words in Manipulative Comments.

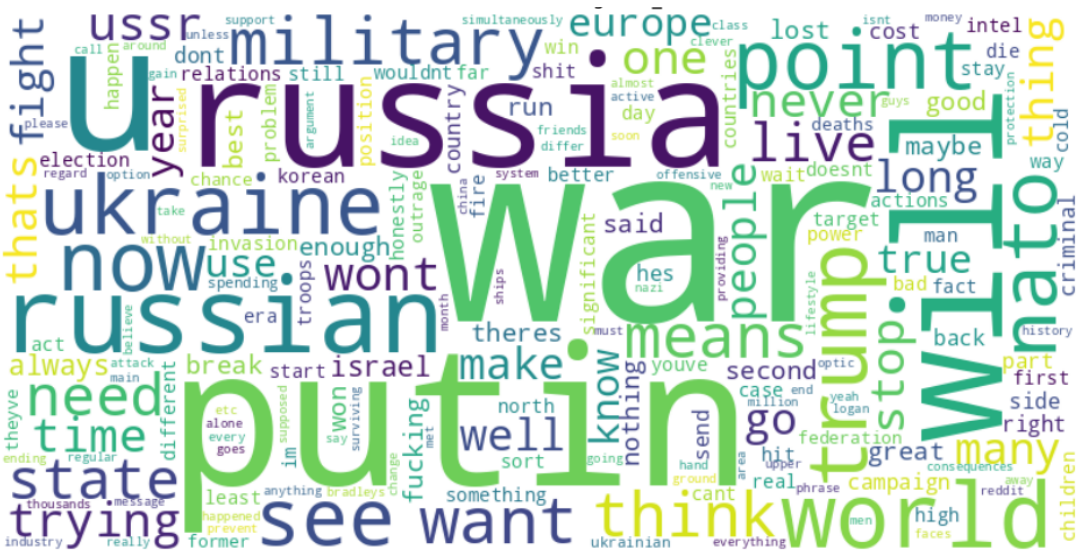

Figure 9: The most common words against the results of Russia's politics and war.

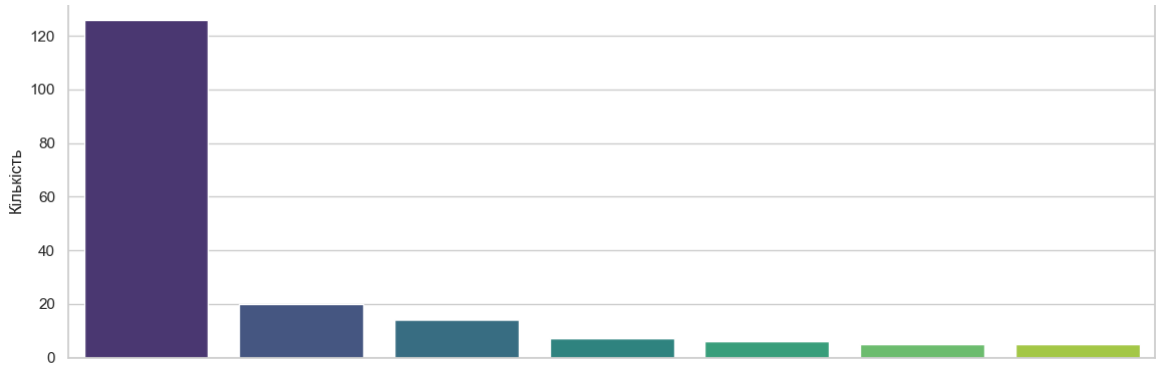

Figure 10: The number of manipulations by type, where generalisation and polarisation, legitimisation of actions, discrediting, distortion of facts, propaganda vocabulary, emotional manipulation and use of labels are left to the right.

With all these analyses and many others, a model has been trained to identify the type of manipulation and the type of political view. The Gradio interface allows you to enter a comment and get:

1. Political position: Pro-Ukraine, Pro-Russia, Neutral.
2. Translation (if the comment is not in English).
3. Words that signal manipulation.
4. Types of manipulation, if detected (emotional, propagandistic, etc.).

The entire text goes through the following stages:

1. Cleanup – removal of URLs, tags, punctuation.
2. Vectorisation – conversion of input text to TF-IDF.
3. Classification – the LogisticRegression or SVC model betrays a political position.
4. Detection of manipulations – keywords from a pre-created dictionary.
5. Visualisation of the result – Markdown stylisation with emojis.

The input field is a text form (4 lines) for entering a comment (Fig. 11). Output result:

1. Political position, for example, the Pro-Ukraine classification.
2. Translation (if any): "Glory to Ukraine! Putin is a war criminal.".
3. Types of manipulation: ["emotional manipulation", "use of labels"].
4. Tags: ["glory", "war criminal"].

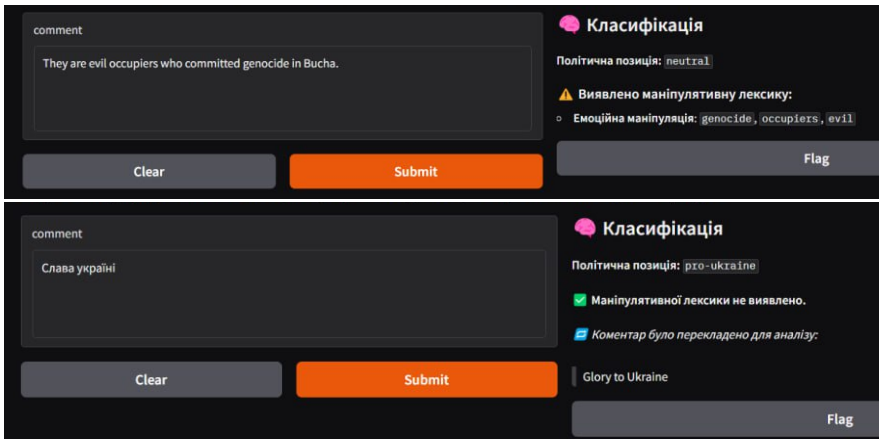

Figure 11: Example of use for an English-language commentary and an example of use for a Ukrainian-language commentary.

8. Program Execution Statistics

As part of our project, we have collected more than 6,000 Reddit comments. Of these:

1. 5453 had a classification of neutral_or_third_party.
2. 297 – pro_ukraine.
3. 250 – pro_russia.

To prepare the dataset, the text was cleaned, noise removal was performed, and non-English comments were translated into English. Afterwards, the texts were vectorised using the SentenceTransformer model (multilingual-e5-base). To separate the sample, the following are used:
train_test_split(X, y, test_size=0.2, random_state=42, stratify=y_encoded)
Model is LogisticRegression(max_iter=1000, class_weight="balanced"). Metrics show in Fig. 12:

1. Neutral: precision = 0.80, recall = 0.92, f1-score = 0.86.
2. Pro-Ukraine: precision = 0.83, recall = 0.62, f1-score = 0.71.
3. Pro-Russia: The model exhibits very low recall and precision, as it fails to capture patterns due to the limited number of examples.

	precision	recall	f1-score	support
neutral	0.80	0.92	0.86	26
pro-russia	0.00	0.00	0.00	2
pro-ukraine	0.83	0.62	0.71	16
accuracy			0.77	44
macro avg	0.54	0.52	0.52	44
weighted avg	0.78	0.77	0.77	44

Figure 12: Metrics.

The model shows a strong classification for neutral, a medium classification for pro_Ukraine, and a weak classification for pro_Russia. It actually suggests that a small percentage of people hold a pro-Russian stance. The problem of class imbalance is a key issue. Comments are grouped by vector proximity (Fig. 13). English speakers form a dense cluster. Other languages are dispersed. It is because the platform is still more popular in English-speaking countries. Additionally, people sometimes write posts in English so that they are understandable to the public. Distribution of the envisaged political positions (Fig. 14):

1. neutral_or_third_party: 5453.
2. pro_ukraine: 297.
3. pro_russia: 250.

Reddit is a platform dominated by neutral or moderate discourse. Often, users discuss the conflict but do not take a clear side. It may be due to Reddit's predominantly English-speaking audience, which tends to be more of an observer than an active participant in events. In addition, many users express analytical, satirical or ironic views, which are difficult to attribute to a clear position. Tonality distribution among the types of manipulations (Fig. 15–16):

1. The use of labels is most associated with negative tonality (≈60%).
2. Emotional manipulation often occurs in neutral and positive comments.
3. Propaganda vocabulary is in pro-Ukrainian posts.

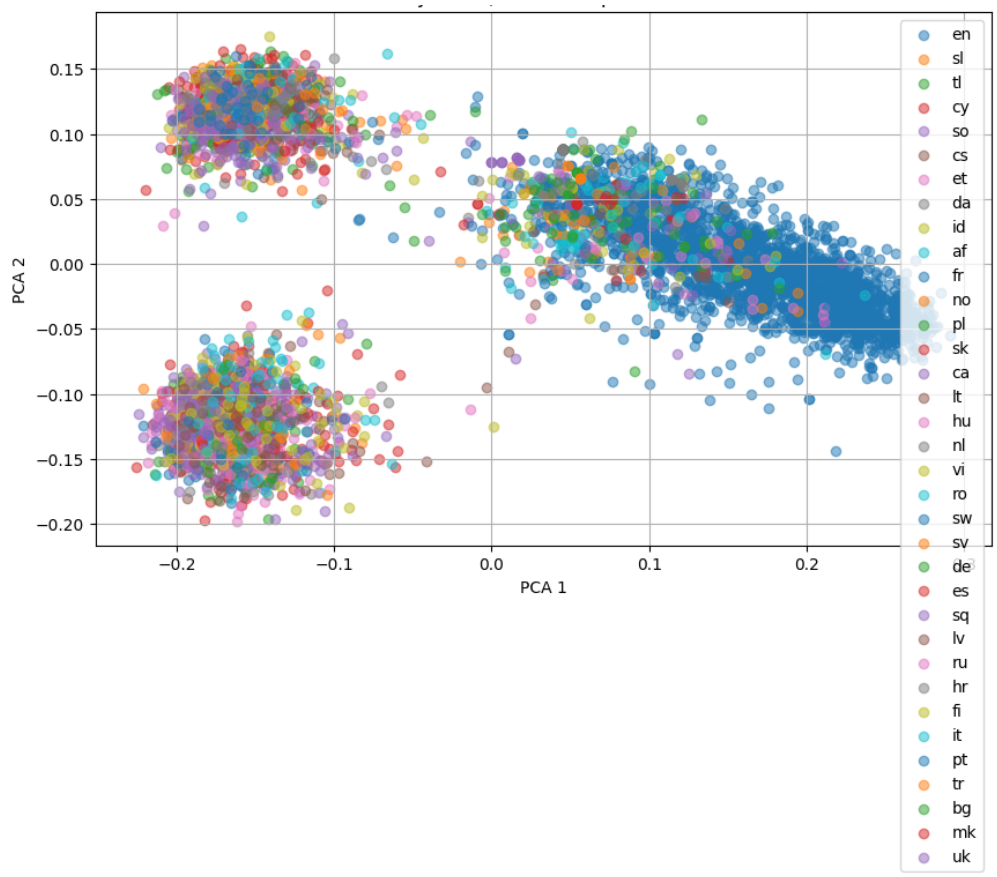

Figure 13: PCA visualisation of the comment vector space by languages (number of languages in comments).

```
stance_prediction
neutral_or_third_party    5453
pro_ukraine                297
pro_russia                 250
Name: count, dtype: int64
```

Figure 14: Foresight.

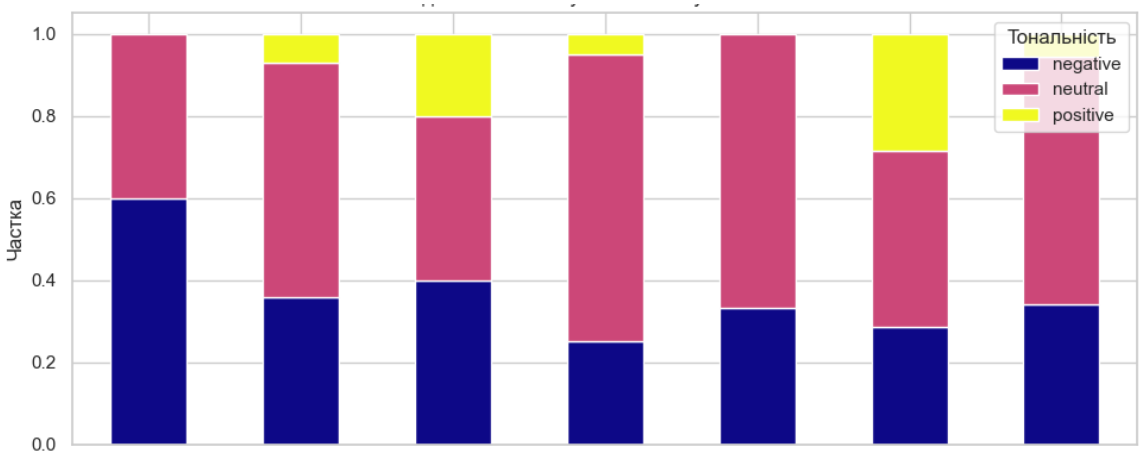

Figure 15: Distribution of tonality among types of manipulative vocabulary, where from left to right the use of labels, discretisation, emotional manipulation, legitimisation of actions, propaganda vocabulary, distortion of facts, generalisation and polarisation.

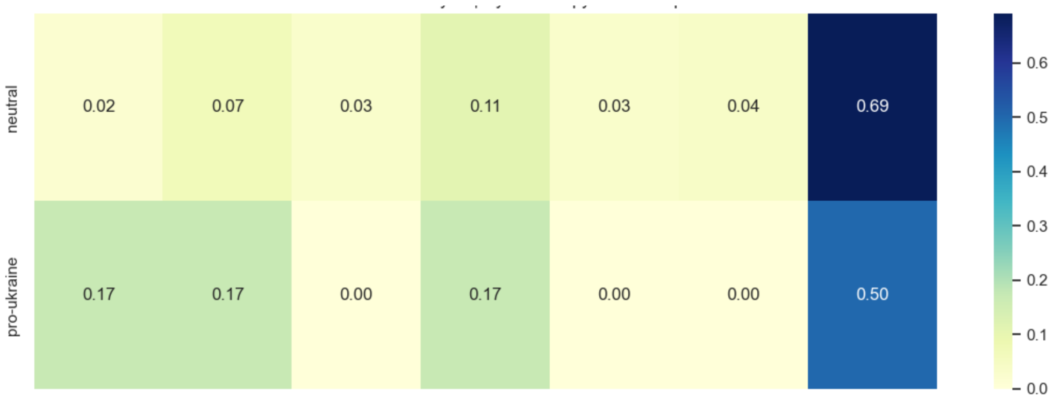

Figure 16: Share of types of manipulation in each group of comments on political position, where from left to right the use of labels, discretisation, emotional manipulation, legitimisation of actions, propaganda vocabulary, distortion of facts, generalisation and polarisation.

The type of manipulation often correlates with the political position. Emotional and propaganda language is used to mobilise or strengthen support, while labels and discrediting are employed against the enemy. It suggests the possibility of building a hybrid classification model that combines political position and type of manipulation. In particular, manipulation in the form of "labelling" (for example, "terrorists", "aggressors") is a marker of hostile or radical discourse. Frequency analysis: keywords (Fig. 17–19):

1. Neutral: russian, Trump, kgb, brigade, office.
2. Pro-Ukraine: Ukraine, freedom, slava, support, hope.
3. Pro-Russia: russia, real, protect, territory, help.

Neutral comments often contain technical, informative vocabulary without vivid emotionality. It explains the high accuracy of the model when classifying this class. Pro-Ukrainian texts tend to use emotionally colored words of support and national elevation. Pro-Russians, on the other hand, focus on justifying actions or defending, which indicates a defensive tone. All these features of vocabulary can serve as good indicators for identifying latent positions in social discussions. Bigrams (related words, Fig. 20):

1. Neutral: don't know, looks like, nuclear weapons, Middle East.
2. Pro-Ukraine: support Ukraine, war ukraine, slava ukraini.
3. Pro-Russia: russian soldiers, russian propaganda, black sea.

Figure 17: Word Cloud for Neutrals.

Figure 18: Word cloud for pro-Ukraine.

Figure 19: Word cloud for pro-russia.

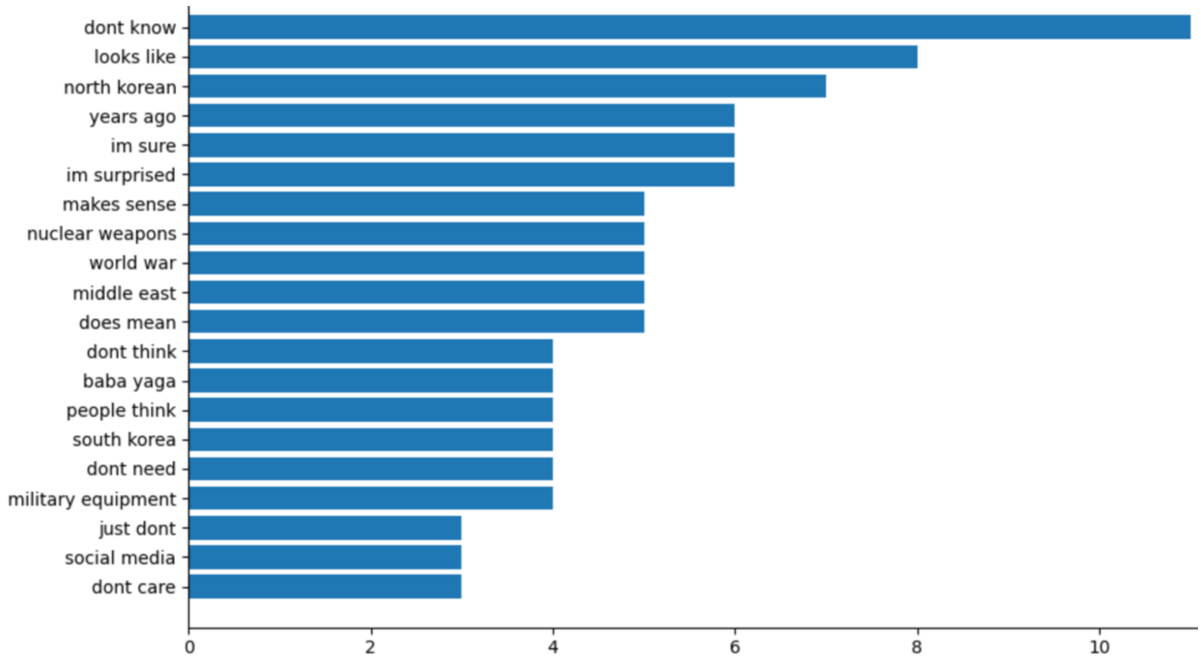

Figure 20: The most common bigrams in neutral comments.

Bigrams enhance interpretation, providing additional semantics for contextual analysis. And in fact, they show us that neutrality sometimes comes either from ignorance or simply from the acceptance of other parties. The model of classifying Reddit comments by political position and manipulative vocabulary showed stable results when using modern transformers (multilingual-e5-base). A strong imbalance of classes was revealed, which was partially compensated for by the use of class_weight=balanced. Surveillance:

1. Neutral rhetoric dominates – Reddit's audience is prone to rational analysis and factual presentation.
2. Manipulative vocabulary does not always accompany extreme positions, but it is more likely to occur in such contexts.
3. The use of specific vocabulary and phrases (bigrams) enables the identification of ideological patterns even with a superficial analysis.

9. Conclusions

During the study, a multilingual information system was developed and tested to analyse the political stance, tone, and presence of manipulative language in comments by Reddit users regarding Russia's war against Ukraine. The implemented approach demonstrated that the combination of modern NLP methods, multilingual models, and manual data markup provides high efficiency in analysing complex online political discourse.

The results confirmed that Reddit is a significant source for international opinion research, as it contains a large volume of emotionally charged, controversial, and politically oriented statements. The multilingual E5-base model demonstrated the ability to adequately vectorise texts in three languages, enabling a unified analysis across all language groups.

The system completed the following tasks:

1. Cleaning and normalisation of texts.
2. Determining the language and tone of comments.
3. Thematic clustering of discourse.
4. Heuristic and machine classification of political position (Pro-UA, Pro-RU, Neutral).
5. Identification of manipulative and propagandistic narratives through a dictionary approach.
6. Building visual analytical models (word clouds, PCA visualisations, frequency distributions).

The analysis revealed that a substantial portion of politically charged commentary employs characteristic terminology associated with propaganda frames. Thematic clusters, meanwhile, highlight the most contentious aspects of the war, including weapons, refugees, geopolitical motives, humanitarian consequences, and manipulative narratives. The use of integrated linguistic and statistical processing has proven effective in identifying hidden trends and understanding the influence of information.

At the same time, the study revealed several limitations, including dependence on the quality of manual markup, the difficulty in interpreting ironic and ambiguous statements, as well as the limited availability of data in Ukrainian and Russian. Additionally, the system lacks mechanisms for distinguishing between bot-generated comments and does not operate in real-time.

Overall, the results demonstrate that the developed system is an effective tool for analysing the information space during wartime and can be used to monitor public sentiment, counter disinformation, and identify political narratives on social networks. The findings are of practical value to analysts, researchers, journalists, and institutions in the fields of education, government, and information security.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] S. M. Mohammad, 9 - sentiment analysis: Detecting valence, emotions, and other affectual states from text, Emotion Measurement (2016) 201–237. doi:10.1016/B978-0-08-100508-8.00009-6.

[2] C. Hutto, E. Gilbert, VADER: A parsimonious rule-based model for sentiment analysis of social media text, in: Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media, The AAAI Press, Palo Alto, CA, 2014, pp. 216–225. doi:10.1609/icwsm.v8i1.14550.

[3] S. Bird, E. Klein, E. Loper, Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit, O'Reilly Media, Inc. Sebastopol, CA, 2009.

[4] R. Baly, G. Karadzhov, A. Saleh, J. Glass, P. Nakov, Multi-task ordinal regression for jointly predicting the trustworthiness and the leading political ideology of news media, arXiv preprint arXiv:1904.00542 (2019). doi:10.48550/arXiv.1904.00542.

[5] Asaniczka, Public opinion Russia Ukraine war (kaggle dataset), 2025. URL: https://www.kaggle.com/datasets/asaniczka/public-opinion-russia-ukraine-war-updated-daily

[6] Reddit, Reddit API documentation, 2025. URL: https://www.reddit.com/dev/api/

[7] A. Zubiaga, A. Aker, K. Bontcheva, M. Liakata, R. Procter, Detection and resolution of rumours in social media: A survey, ACM Computing Surveys 51 (2) (2018) 1–36. doi:10.1145/3161603.

[8] A. Hanselowski, A. PVS, B. Schiller, F. Caspelherr, D. Chaudhuri, C. M. Meyer, I. Gurevych, A retrospective analysis of the fake news challenge stance detection task, arXiv preprint arXiv:1806.05180 (2018). doi:10.48550/arXiv.1806.05180.

[9] A. Hanselowski, I. Gurevych, A framework for automated fact-checking for real-time validation of emerging claims on the web, in: Proceedings of the Workshop on Prioritising Online Content, NeurIPS Foundation, Long Beach, CA, 2017, pp. 1–3.

[10] A. Hanselowski, C. Stab, C. Schulz, Z. Li, I. Gurevych, A richly annotated corpus for different tasks in automated fact-checking, arXiv preprint arXiv:1911.01214 (2019). doi:10.48550/arXiv.1911.01214.

[11] S. Volkova, K. Shaffer, J. Y. Jang, N. Hodas, Separating facts from fiction: Linguistic models to classify suspicious and trusted news posts on Twitter, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Association for Computational Linguistics, Stroudsburg, PA, 2017, pp. 647–653. doi:10.18653/v1/P17-2102.

[12] S. Volkova, E. Ayton, D. L. Arendt, Z. Huang, B. Hutchinson, Explaining multimodal deceptive news prediction models, in: Proceedings of the Thirteenth International AAAI Conference on Web and Social Media, AAAI Press, Palo Alto, CA, 2019, pp. 659–662. doi:10.1609/icwsm.v13i01.3266.

[13] S. M. Mohammad, P. Sobhani, S. Kiritchenko, Stance and sentiment in tweets, TOIT 17 (3) (2017) 1–23. doi:10.1145/3003433.

[14] A. Hanselowski, H. Zhang, Z. Li, D. Sorokin, B. Schiller, C. Schulz, I. Gurevych, UKP-athene: Multi-sentence textual entailment for claim verification, arXiv preprint arXiv:1809.01479 (2018). doi:10.48550/arXiv.1809.01479.

[15] K. Shu, S. Wang, D. Lee, H. Liu, Mining disinformation and fake news: Concepts, methods, and recent advancements, in K. Shu, S. Wang, D. Lee, H. Liu (Eds.), Disinformation, Misinformation, and Fake News in Social Media: Emerging Research Challenges and Opportunities, Springer, Cham, Switzerland, 2020, pp. 1–19. doi:10.1007/978-3-030-42699-6_1.

[16] N. Kotonya, F. Toni, Explainable automated fact-checking for public health claims, arXiv preprint arXiv:2010.09926 (2020). doi:10.48550/arXiv.2010.09926.

[17] S. Volkova, M. Glenski, E. Ayton, E. Saldanha, J. Mendoza, D. Arendt, Z. Shaw, K. Cronk, S. Smith, M. Greaves, Machine intelligence to detect, characterise, and defend against influence operations in the information environment, Journal of Information Warfare 20 (2) (2021) 42–66.

[18] W. Medhat, A. Hassan, H. Korashy, Sentiment analysis algorithms and applications: A survey, Ain Shams Engineering Journal 5 (4) (2014) 1093–1113. doi:10.1016/j.asej.2014.04.011.
[19] J. Devlin, M. W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conferance of the North American Chapter of the ACL: Human Language Technologies, Association for Computational Linguistics, Stroudsburg, PA, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423.
[20] S. Shen, J. Liu, L. Lin, Y. Huang, L. Zhang, C. Liu, Y. Feng, D. Wang, SsciBERT: A pre-trained language model for social science texts, Scientometrics 128 (2023) 1241–1263. doi:10.1007/s11192-022-04602-4.
[21] V. Vysotska, K. Przystupa, Y. Kulikov, S. Chyrun, Y. Ushenko, Z. Hu, D. Uhryn, Recognizing fakes, propaganda and disinformation in Ukrainian content based on NLP and machine-learning technology, IJCNIS 17 (1) (2025) 92–127. doi:10.5815/ijcnis.2025.01.08.
[22] V. Vysotska, Computer linguistic system architecture for Ukrainian language content processing based on machine learning, in: Proceedings of the Modern Data Science Technologies Workshop, MoDaST '2024, CEUR Workshop Proceedings, Aachen, Germany, 2024, pp. 133–181.
[23] S. Mainych, A. Bulhakova, V. Vysotska, Cluster analysis of discussions change dynamics on Twitter about war in Ukraine, in: Proceedings of the 7th International Conference on Computational Linguistics and Intelligent Systems. Volume II: Computational Linguistics Workshop, CoLInS '2023, CEUR Workshop Proceedings, Aachen, Germany, 2023, pp. 490–530.
[24] V. Vysotska, Computer linguistic systems design and development features for Ukrainian language content processing, in: Proceedings of the 8th International Conference on Computational Linguistics and Intelligent Systems. Volume III: Intelligent Systems Workshop, ISW-CoLInS '2024, CEUR Workshop Proceedings, Aachen, Germany, 2024, pp. 229–271.
[25] V. Vysotska, M. Nazarkevych, S. Vladov, O. Lozynska, O. Markiv, R. Romanchuk, V. Danylyk, Devising a method for detecting information threats in the Ukrainian cyber space based on machine learning, Eastern-European Journal of Enterprise Technologies 132 (2) (2024). doi:10.15587/1729-4061.2024.317456.
[26] V. Vysotska, A. Chupryna, N. Valenda, O. Konduforov, Evaluating the in-context learning capabilities of large language models for misinformation detection for Ukrainian news, in: Proceedings of the Computational Intelligence Application Workshop, CIAW '2025, CEUR Workshop Proceedings, Aachen, Germany, 2025, pp. 181–197.
[27] I. Peleshchak, V. Lytvyn, V. Vysotska, O. Khobor, M. Luchkevych, A lightweight cross-attentive Tinybert with LoRA and Bi GRU for fake news detection, in: Proceedings of the Computational Intelligence Application Workshop, CIAW '2025, CEUR Workshop Proceedings, Aachen, Germany, 2025, pp. 29–43.
[28] V. Vysotska, M. Nazarkevych, Development of an information technology for detecting the sources and networks of disinformation dissemination in cyberspace based on machine learning methods, Eastern-European Journal of Enterprise Technologies 4 (2025). doi:10.15587/1729-4061.2025.335501.
[29] V. Vysotska, L. Chyrun, S. Chyrun, I. Holets, Information technology for identifying disinformation sources and inauthentic chat users' behaviours based on machine learning, in: Proceedings of the Modern Data Science Technologies Workshop, MoDaST '2024, CEUR Workshop Proceedings, Aachen, Germany, 2024 pp. 427-465.
[30] V. Vysotska, P. Pukach, V. Lytvyn, D. Uhryn, Y. Ushenko, Z. Hu, Intelligent analysis of Ukrainian-language tweets for public opinion research based on NLP methods and machine learning technology, IJMECS 15 (3) (2023) 70–93. doi:10.5815/ijmecs.2023.03.06.