I.J. Image, Graphics and Signal Processing, **2025, 5, 117-164**
Published Online on October 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2025.05.08

Sentiment Analysing and Visualising Public Opinion on Political Figures across YouTube and Twitter Using NLP and Machine Learning

Victoria Vysotska
Department of Information Systems and Networks, Lviv Polytechnic National University, Lviv, 79013, Ukraine
E-mail: Victoria.A.Vysotska@lpnu.ua
ORCID iD: https://orcid.org/0000-0001-6417-3689

Alina Starchenko
Department of Information Systems and Networks, Lviv Polytechnic National University, Lviv, 79013, Ukraine
E-mail: alina.starchenko.sa.2022@lpnu.ua
ORCID iD: https://orcid.org/0009-0004-1360-2948

Lyubomyr Chyrun
Ivan Franko National University of Lviv, Lviv, 79000, Ukraine
E-mail: Lyubomyr.Chyrun@lnu.edu.ua
ORCID iD: https://orcid.org/0000-0002-9448-1751

Zhengbing Hu
School of Computer Science, Hubei University of Technology, Wuhan, China
E-mail: drzbhu@gmail.com
ORCID iD: https://orcid.org/0000-0002-6140-3351

Yuriy Ushenko*
Department of Computer Science, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, 58012, Ukraine
E-mail: y.ushenko@chnu.edu.ua
ORCID iD: https://orcid.org/0000-0003-1767-1882
*Corresponding Author

Dmytro Uhryn
Department of Computer Science, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, 58012, Ukraine
E-mail: d.ugryn@chnu.edu.ua
ORCID iD: https://orcid.org/ 0000-0003-4858-4511

Received: 12 March, 2025; Revised: 22 May, 2025; Accepted: 09 July, 2025; Published: 08 October, 2025

Abstract*:* The study is devoted to the analysis of public sentiment towards Ukrainian political figures based on comments on social media, in particular, YouTube and Twitter. The work aims to identify differences in the perception of political leaders and to understand how the platform affects the tone of statements. The main research question is to determine how public opinion about politicians in Ukraine differs between YouTube and Twitter during the full-scale war. To do this, a corpus of comments and tweets from 2022 to 2023 was collected, which went through pre-processing stages (including cleaning up slang and spelling mistakes). The article presents the results of a comprehensive analysis of public opinion on five public figures of Ukraine (S. Prytula, P. Poroshenko, V. Zelensky, S. Sternenko, A. Yermak) based on data from the social networks YouTube and Twitter. For data collection, the YouTube Data API and the Apify platform were used, a corpus of Ukrainian-language comments and tweets was collected and processed, which went through the stages of purification, normalisation and lemmatisation, taking into account slang, surzhyk and spelling mistakes. The sentiment analysis model, built on the basis of multilingual-e5-base embeddings and the XGBClassifier algorithm, showed an accuracy of 89.4%, macro-F1 of 88.7%, and a weighted F1 of 89.1%. Sentiment distribution analysis revealed that, on average, 42% of messages were positive, 36% were negative, and 22% were neutral. Twitter had a higher share of negative statements (up to 40%), while YouTube had a predominance of positive sentiment (up to 47%). The results indicate differences in the perception of public figures on different platforms and confirm the

effectiveness of the developed approach for the Ukrainian-speaking segment of social networks. The results indicate significant differences in sentiment distribution: comments on YouTube are more likely to be marked by emotional intensity and harshness. At the same time, Twitter exhibits a more concise but no less polarised discourse. One of the reasons for this difference may be the difference in the format of the platforms, their audience, and the speed of content distribution. Further research should take into account the impact of user demographic biases, as well as the activity of bots or coordinated campaigns that can change the perception of public opinion. The practical significance of the study lies in the fact that its results can be used by politicians, journalists, and public figures to better understand the mood of society, predict reactions to political events, and build more effective communication. At the same time, it is worth noting that there are limitations: automated sentiment analysis has difficulty detecting sarcasm, irony, or context-sensitive meanings, which can affect the Accuracy of the results. In addition, the study takes into account the ethical aspects of data collection and analysis: only publicly available comments were used, without interference in the private sphere of users. There are possible risks of abuse of such technologies, and the need for responsible application of the findings is emphasised.

Index Terms: Sentiment Analysis, Public Opinion, Social Networks, Twitter, Youtube, Ukrainian-Language Content, Natural Language Processing, NLP, Machine Learning

1. Introduction

In today's world, social networks have become almost indispensable tools for communication, exchange of opinions, and shaping public sentiment. Vital platforms are Twitter (now X) and YouTube, which not only allow users to freely express their views but also allow researchers to record and analyse these statements. Twitter is characterised by short, prompt messages that will enable you to quickly track reactions to events, while YouTube provides the ability to analyse more detailed comments under video content. An exciting direction is the study of public perception by society of public figures - politicians, public figures, and opinion leaders, since their public perception directly affects political and social processes [1]. The relevance of the study is due to the fact that public opinion, formed and presented on the Internet, is increasingly becoming a real driving force for changes in the adoption of laws or public management decisions in public or private institutions. Thanks to the activity of users in social networks, important topics are discussed, public discussions are initiated, investigations into the activities of public figures are launched, and sometimes even the course of political processes is changed [2]. Examples of the authorities' reaction to public outcry on various platforms show that the opinion of ordinary citizens can influence the so-called "already made" decisions, stop appointments, speed up or stop the consideration of draft laws, or provoke official statements. In wartime, the growth of social tension and restrictions or even a ban on peaceful assemblies [3] has only increased this impact on the strength of opinions expressed regarding the activities of officials, volunteers and public figures. For this study, five public figures from Ukraine were selected: Serhiy Prytula, Petro Poroshenko, Volodymyr Zelensky, Serhiy Sternenko, and Andriy Yermak. These people are politicians and public activists, which allows us to get a comprehensive picture of public sentiment towards public figures in various fields of activity. Another sign of relevance is the insufficient study of the Ukrainian-language segment of social networks, especially on Twitter and YouTube. Tools for sentiment analysis either do not support the Ukrainian language or give incorrect results due to the limited language models for working with Ukrainian [4]. Therefore, the study, focused on the analysis of Ukrainian data from these two key platforms, has not only practical but also scientific value. Due to the constant growth in the volume of online discussions, the need for automated tools that can quickly and efficiently process large amounts of text data is also growing. For this task, natural language processing (NLP) and machine learning methods become especially valuable, because they allow not only to determine the tone of statements, but also to make this process automated and more efficient. So, the chosen topic is very relevant both in scientific and practical directions, as it combines modern methods of text analysis with a socially significant object – public figures in the context of the information space of Ukraine.

In the modern information space, social networks have become a key communication channel where public opinion is formed and broadcast. Twitter (X) and YouTube occupy a special place as platforms that combine quick emotional reactions and lengthy user comments. In this context, automated technologies are needed to analyse Ukrainian-language content, which would allow a timely assessment of the mood of society towards political and public figures.

The purpose of the study is to develop and test innovative technology for collecting, processing and analysing Ukrainian-language texts from Twitter and YouTube in order to identify and model public opinion regarding well-known Ukrainian public figures. Several aspects determine the difference from existing solutions:

– Focus exclusively on the Ukrainian language, including surzhyk, slang and spelling mistakes, which is not taken into account by most available platforms;

– A multi-platform approach (Twitter and YouTube), which combines short emotional statements with more reasoned comments;

– Creation of a specialised dictionary of Ukrainian Internet slang for political discourse;
– Use of modern natural language processing models (multilingual-e5-base, XGBClassifier) and integrate them into the application cycle from data collection to interactive visualisation.

The main limitations of the proposed technology are:

– The analysis covers only two social networks, which can narrow the representativeness of the results.
– Automatic detection of sarcasm, irony and hidden meanings remains a difficult task.
– It is not always possible to eliminate the influence of bot accounts and artificially generated content.
– The results reflect sentiment only in a specific time period.

The aim of this study is to identify and analyse the sentiment of public opinion towards public figures in Ukraine by collecting, processing and analysing text data from social networks using NLP adapted to the political and social context of Ukraine. Research objectives:

- Formation of a sample of public figures for analysis.
- Configure access to the Twitter and YouTube APIs to retrieve data.
- Receiving tweets from Twitter users mentioning selected public figures and comments from YouTube under video interviews with these individuals.
- Marking the resulting corpus by dividing all data into negative, neutral, and positive statements.
- Text pre-processing, namely data cleaning, normalisation, noise removal (spam, duplicates, non-informative texts) and embedding.
- Development and adaptation of the sentiment analysis model for Ukrainian-language content on the formed corpus.
- Visualisation of the results in the form of graphs that reflect the attitude of the public towards public figures, as well as graphs to depict sentiment for each of the platforms.
- Formulating conclusions on general trends in public opinion and the potential of using analysis to predict public sentiment.

The object of the study is the process of formation and manifestation of public opinion in social networks, Twitter and YouTube, which act as a reflection of public moods and opinions, allowing them to be used as a source of data for analysing the public perception of various social or political figures. The subject of the study is the statements of Ukrainians about public figures on Twitter and YouTube, as well as the analysis of tweets and comments in Ukrainian that mention elected politicians and public figures, to understand the public's emotional colouring towards them. It is also taken into account that people communicate differently on Twitter and in YouTube comments.

The scientific novelty lies in the development of a complete applied cycle of analysis of public sentiment in the Ukrainian-language segment of social networks Twitter and YouTube from data collection to interactive visualisation using NLP and machine learning methods.

- The study focuses on the Ukrainian-language segment of selected social networks, which is often left out of the attention of global analysis models.
- A dictionary of Ukrainian Internet slang was created and adapted to political discourse, followed by automated replacement of such words in texts.
- Conducting a comparative analysis of sentiment features between Twitter messages and YouTube comments.
- Comparing the performance of several machine learning models for the sentiment classification problem for comparison and selection of the best model.
- Creation of an interactive interface for visualising the results of analysis in Streamlit using Seaborn and Matplotlib libraries, which allows you to visually track public sentiment towards individual public figures.

Thus, the study combines work with Ukrainian-language informal texts from two different platforms, modern NLP approaches, manual labelling and visual analytics, which makes it relevant and valuable in the context of public opinion analysis. The study creates a fundamental tool that can be used to understand what society thinks about public figures based on their activity on Twitter and YouTube. This information will be necessary for many areas.

- Political analysts will be able to track the rating of trust in public figures.
- Journalists will react faster to information waves and analyse which topics evoke more emotions in Ukrainians.
- Sociologists will get a new way to study public opinion without conducting expensive surveys.
- PR specialists will be able to notice problems with the reputation of their clients in time.

Based on the study, it is possible to create a website or application with graphs and diagrams of people's attitudes towards various public figures. By focusing on two key platforms (Twitter and YouTube), it will be possible to get a

complete picture of what Ukrainians think about their public figures, since people express themselves differently in tweets and comments under the video.

Practical significance of the approach:

1. **Media monitoring.** The developed approach can be used to automatically track public sentiment in near real-time. It allows journalists and media analysts to:

– Quickly detect changes in attitudes towards political figures,

– Distinguish between waves of support and criticism,

– Track information campaigns formed on social networks.

2. **Political strategy.** For policy-makers and their teams, the tool can be a source of:

– feedback on the perception of speeches, statements or decisions,

– comparison of reputation between different platforms (for example, where it is worth investing more resources in communication, in YouTube or Twitter),

– prompt adaptation of rhetoric to the mood of voters.

3. **Crisis communication.** In cases of crisis events (scandal, military news, social protests), the tool allows:

– identify "outbursts" of negative moods,

– evaluate the effectiveness of official statements,

– promptly respond to reputational threats or the spread of disinformation.

According to the structure of the article, section 2 provides a review of the literature and related works. Section 3 formulates the statement of the problem and substantiates the difference from existing solutions. Section 4 describes the methodology for data collection and pre-processing. Chapter 5 provides a mathematical scheme for training models. Chapter 6 presents the results of implementation and visualisation. In the final sections, conclusions are formulated, limitations are identified, and directions for further research are outlined.

2. Related Works

Analysis of public opinion based on textual content from social networks is an active area of research in the fields of natural language processing (NLP), sociology, and political science. A significant number of works are focused on high-resource languages, while the Ukrainian-language segment has remained insufficiently researched for a long time [5-6]. Traditional lexicon approaches, such as VADER or SentiStrength, are often used to analyse short Twitter posts or YouTube comments [7-8]. However, they demonstrate low accuracy in working with the Ukrainian language due to ignoring slang, surzhyk and the context of political discourse. Studies [9-10] show that adapting dictionaries or creating new corpora for local languages significantly improves the quality of tonality classification. In recent years, there has been a transition to transformer models, particularly XLM-RoBERTa, mBERT and specialised models for the Ukrainian language (for example, Ukr-RoBERTa) [11]. Models further trained on Ukrainian-language corpora significantly outperform classical machine learning methods (SVM, XGBoost, Naïve Bayes) in sentiment analysis problems [12].

An important area is multiplatform analysis. Research [13] shows that Twitter posts are more likely to be short, emotionally rich, and contain a higher percentage of negative ratings. At the same time, YouTube comments are more detailed, reasoned, and often neutral. IIIe justifies the need for comparative analysis to obtain a more complete picture of public sentiment. A number of papers also point to issues of representativeness and noise in data [14-15], in particular, the impact of bot accounts and automatically generated content on results. Research [16] demonstrates that filtering out suspicious accounts and handling irony/sarcasm can significantly alter the final sentiment scores.

Thus, modern research confirms the effectiveness of using transformer models adapted to the Ukrainian context in combination with multiplatform data, while highlighting the need to further improve methods for handling sarcasm, slang and bot detection.

Despite the comprehensive approach, the study has a number of limitations. First, the corpus of data is formed from only two platforms (Twitter and YouTube), which may not reflect the full range of public opinion on other social networks. Secondly, in the process of data collection, it is not always possible to completely exclude automatically generated or bot content that can affect the results. Thirdly, the problem of automatic recognition of sarcasm and irony remains unresolved, which can lead to a misclassification of emotional tonality. Also, a limitation is the time sample – the analysis reflects moods in a specific period and does not take into account their dynamics over a long time. It is advisable to focus further research on:

- Expansion of data sources – including Facebook, Instagram, Telegram, and forums to increase representativeness.
- Detection of bots and information attacks – development of modules for automatic detection and filtering of artificially generated content.
- The analysis of sarcasm and irony is the integration of specialised models trained in political discourse.
- Dynamic analysis – building time series and identifying changes in public mood in the context of events.
- The multimodal approach takes into account not only the text but also the video and pictorial content that accompany the comments.

We will carry out a thorough analysis of such studies (comparison of Twitter vs YouTube), with references to key works and conclusions about the strengths/weaknesses of the approaches. In recent years, in the field of public opinion analysis and sentiment analysis, the number of works aimed at low-resource languages (including Ukrainian) and the study of public sentiment in the context of significant events (for example, war) has increased significantly. Currently, some works create specialised datasets for Ukrainian (emotions/sentiment) and benchmarks for evaluating models [17]. The main typical approaches include lexicon methods (VADER, SentiStrength), classical ML (SVM, XGBoost) and modern transformers (XLM-RoBERTa, Ukr-RoBERTa, multilingual-E5, etc.). For Ukrainian, the change in balance in favour of transformers is confirmed by comparative studies: transformer frameworks give much higher accuracy in tonality/emotions, provided that there are marked data or correct additional training [18-19]. Studies comparing YouTube tweets and comments show that while cross-network communities can correlate (users of the same community are active on both platforms), the message genres themselves differ – tweets are shorter, faster, with a greater share of negativity/polarisation; under YouTube videos, longer, often more reasoned or neutral comments. Ще affects the distribution of sentiment and the effectiveness of different models/lexicons. In this regard, the comparative multiplatform approach gives a more complete picture of public opinion [20]. For the Ukrainian language, there is still a lack of wide, diverse and open datasets with correct emotion/polarity labels; Therefore, recent works are actively trying to fill this gap (new benchmarks, synthetic sets, specialised corpus of conflict/political tweets). A large dataset of tweets about the Russian-Ukrainian conflict demonstrates how massive cases help analyse dynamics, but also highlights problems with balance and bots [17, 21]. The most frequently mentioned technical challenges are:

- Language features: surzhyk (dialect), code-switching (ukr/rus/eng), memes and slang complicate normalisation and lexical approaches.
- Irony/sarcasm: automatic recognition remains a weak point.
- Noise (bots, spam): distorts sentiment particles; requires filtering.
- Limited access to historical/full APIs: especially for Twitter, forcing the use of scraping/third-party services [18, 20].

In modern works, accuracy, precision/recall/F1 (macro and weighted), and time/thematic analyses of sentiment dynamics are used. For many Ukrainian-language tasks, additional benchmarks with cross-validation of transfer learning (fine-tuning multilingual models) appear [17]. Our approach (focus on Ukrainian-language data, multiplatform collection, proprietary slang dictionaries, comparison of several ML models, and Streamlit dashboard) solves several key problems: adaptation to local slang/surzhyk, multiplatform comparison, and attention to interactive visualisation of results. It makes the study practice-oriented and representative for the tasks of applied analysis of public opinion. (Support for similar recommendations can be found in recent works on UKR NLP and benchmark datasets.) [17-18].

Recommendations for increasing the weight of results (methodological and reporting) [13]:

- Describe in detail the collection and filtering procedures (to avoid bias due to bots/scraping).
- Provide examples of annotations + instructions to annotators (to increase inter-writer reactivity).
- Show ablation experiments: (a) a model without a slang dictionary, (b) with/without surzhyk normalisation, (c) different vectorisations (TF-IDF vs multilingual-E5) – this will show the contribution of each component.
- Use time-series/event-based sentiment shifts to show practical usefulness.
- Leave open (if possible) a subset of the annotations and the code/dump of the data (anonymised) so that others can replicate the results. (This is how new benchmarks for Ukrainian do) [17].

The field is developing rapidly: transformers with localised datasets give the most significant increase in quality, and multiplatform analysis (Twitter vs YouTube) provides deeper insights about different types of audiences. The main challenges are the resource base for Ukrainians, the processing of slang/sarcasm, and noise (bots). Your approach responds very well to these challenges and can make a significant contribution if you add clear validation experiments and make a piece of data/code public for replication [17, 20-21].

Table 1. Comparison of similar studies in the field of public opinion analysis

[X]	Language of study	Platform	Methods	Dataset	Key results
1	English	Facebook, Twitter	content analysis, descriptive statistics, SVM, Naïve Bayes	Posts and comments from social networks during the political campaign, ~5,000 entries	F1 = 0,78; political campaign in Nigeria. The impact of digital strategies on political participation has been determined
2	Ukrainian, English	Facebook	Social Network Analytics, Statistical Analysis, Lexicon Analysis	Facebook users during the 2019 presidential election in Ukraine, ~15,000 posts	The influence of candidates on the agenda is revealed. Changes in audience activity were detected during the campaign
3	Ukrainian	news sites, social networks	machine learning, NLP, threat detection	collected Ukrainian-language texts from cyberspace, ~10,000 entries	A classification method is proposed with Accuracy = 0.91, F1 = 0.90

4	English, Ukrainian	Text corpora	Semantic-syntactic analysis	Comparative corpus English and Ukr. Texts; ~3,000 texts	Differences in syntax and semantics are shown
5	English	Twitter, Facebook	content analysis, hashtag analysis	data from social networks during the elections in Nigeria; ~20,000 entries	Key themes and campaign vectors identified
6	Ukrainian	Twitter	XGBoost + lexicon	20 thousand. Tweets	F1 = 0,84; Adaptation of the dictionary to the Ukrainian-language discourse
7	English	Twitter	rule-based sentiment analysis (VADER)	Tweets on various topics, ~50,000 tweets	Accuracy = 0.96, F1 = 0.96; effective for short texts. Developed by VADER with high accuracy on short texts
8	English	MySpace, Twitter, YouTube	lexical analysis, Sentiment Strength Detection (SentiStrength, SSD)	short informal texts, ~30,000 messages	Accuracy = 0.84, F1 = 0,74; weak adaptation to sarcasm. An SSD method for measuring the strength of emotions has been proposed.
9	Chinese, English	Chinese media	Linguistic analysis of emotional vocabulary	media texts about Ukraine and the West; ~2,000 articles	A difference in emotional vocabulary regarding Ukraine has been revealed
10	Ukrainian	Twitter	NLP, Machine Learning	Ukrainian-language tweets; ~12,000 entries	Accuracy of classification achieved Accuracy = 0.88, F1 = 0.87
11	Ukrainian	Online reviews	fine-tuning BERT, DistilBERT, XLM-R, Ukr-RoBERTa	corpus of Ukrainian reviews; ~8,000 entries	Ukr-RoBERTa showed the highest Accuracy = 0.92, F1 = 0.92
12	Slovak	Social networks	ML for sentiment analysis	Slovak-language corps; ~5,500 entries	SVM is determined to outperform other models by 4%, Accuracy = 0.86, F1 = 0.85
13	English	YouTube, Twitter, news sites	cross-platform hyperlink analysis	hyperlinks and texts from the media; ~200,000 connections	Cross-media centres of influence identified
14	English	Twitter	bot detection, social network analytics	tweets tagged #IStandWithPutin, #IStandWithUkraine	The significant role of bots in the spread of narratives has been identified
15	Ukrainian, English	News sites	Sarcasm, ML	news headlines; ~10,000 entries	Achieved Accuracy = 0.84, F1 = 0.83 in detecting sarcasm
16	Ukrainian	Texts online	linguistic systems, syntactic analysis	Corps of Ukrainians. Content; ~4,000 texts	A model for the automatic processing of Ukrainian. Texts
17	Ukrainian	Social Networks, News	emotional classification (EmoBench-UA)	Ukrainian benchmark for emotions; ~15,000 entries	A new benchmark for emotional classification is proposed, Accuracy = 0.90, F1 = 0.89
18	Ukrainian	Social networks	ML, sentiment analysis	Corps of Ukrainians. social networks; ~9,000 messages	Achieved Accuracy = 0.90, F1 = 0.89 on the test
19	English	Twitter	XLM-RoBERTa	~10,000 tweets about the metaverse and 6G	Accuracy = 0.93, F1 = 0.93
20	English	Twitter, YouTube	sentiment analysis, graph analysis	data on the 2020 US election; ~50,000 entries	Common patterns were detected between platforms, Accuracy = 0.87, F1 = 0.86
21	English	Twitter	NLP, Analysis of World Discourse	tweets about the Russian-Ukrainian war; ~2 million messages	The world's perception of the conflict is analysed

In the field of opinion analysis, there are already a number of tools and platforms that are used in both scientific research and business. Talkwalker [22] is one of the leading social media monitoring platforms actively used to track the dynamics of public opinion. The platform offers extensive analysis capabilities for sentiment, brand mentions, public figures, and information trends in real-time, with coverage of more than 150 million data sources. Talkwalker formally supports more than 187 languages, including Ukrainian, but this support is superficial. The main limitations for the Ukrainian language are the lack of deep semantic processing of Ukrainian-language content, especially in socio-political discourse, the inability to adequately work with Ukrainian slang, surzhyk, language games and irony, as well as insufficient understanding of culturally determined allusions, which are critical for analysing moods in the public space of Ukraine. In the context of the hybrid language typical of Ukrainian social networks, this significantly reduces the accuracy of interpretations and creates risks of erroneous conclusions.

Sprout Social [23] is a social media monitoring platform with the ability to track brand mentions, analyse user comments, and determine the sentiment of public figures and organisations. The platform serves more than 30,000 brands worldwide and supports various languages, but its primary focus on English-language content limits its effectiveness for analysing Ukrainian-language texts. In particular, Sprout Social's sentiment analysis mechanisms are not able to accurately take into account the nuances of the Ukrainian language, such as surzhyk or specific cultural expressions. An essential limitation for use in the Ukrainian context is also that Sprout Social does not provide flexibility in adding its dictionaries or modifications for specific language constructions. Elements such as irony, sarcasm, and political allusions can be misinterpreted, which is extremely important when analysing public opinion in such a sensitive field as Ukrainian socio-political discourse.

Brandwatch [24] offers advanced tools for analysing mentions of brands, public figures, and key topics with coverage of more than 100 million sources. The platform has powerful capabilities to analyse sentiment, identify trends, and generate reports based on data from various social networks. Brandwatch supports several languages, but like other major platforms, it is mainly focused on English-language content, which limits its effectiveness in working with Ukrainian texts. One of the main problems is that Brandwatch does not take into account the culturally specific

linguistic features of the Ukrainian language, such as surzhyk combinations, local slang expressions, political allusions or irony, which are critical in the analysis of public opinion in Ukraine. Sentiment analysis tools, while working effectively for English-language texts, often fail to provide a correct interpretation when it comes to the Ukrainian context, where the tonality can be significantly distorted due to the use of local language variants.

MonkeyLearn [25] is a text and sentiment analysis tool that provides capabilities for automated classification and extraction of data from text arrays. The platform allows users to create personalised models to detect emotions, categorise texts, and determine sentiment. However, despite its capabilities for analysis, MonkeyLearn has significant limitations when working with the Ukrainian language. Most of the models and algorithms available in MonkeyLearn are focused on English, which leads to low accuracy when processing texts in Ukrainian. A particular problem is the lack of specialised tools for working with the linguistic features of the Ukrainian language, such as surzhyk, informal expressions, local slang expressions and mixing of languages. It poses a risk of misinterpretations, especially when it comes to analysing political or social topics, where language often contains elements of irony and sarcasm.

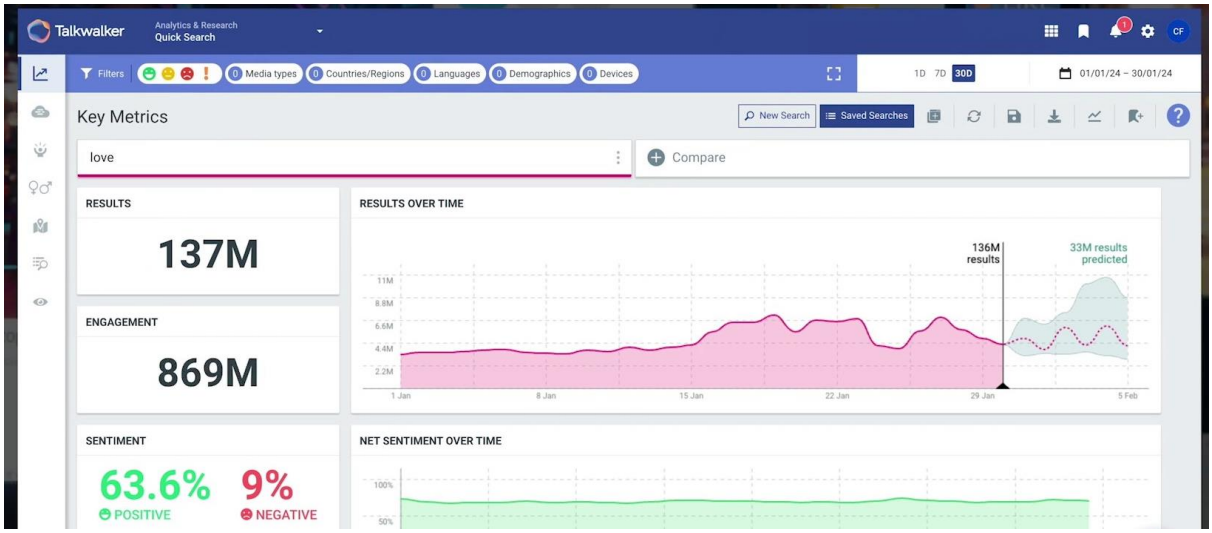

Fig. 1. Example of analysis in Talkwalker

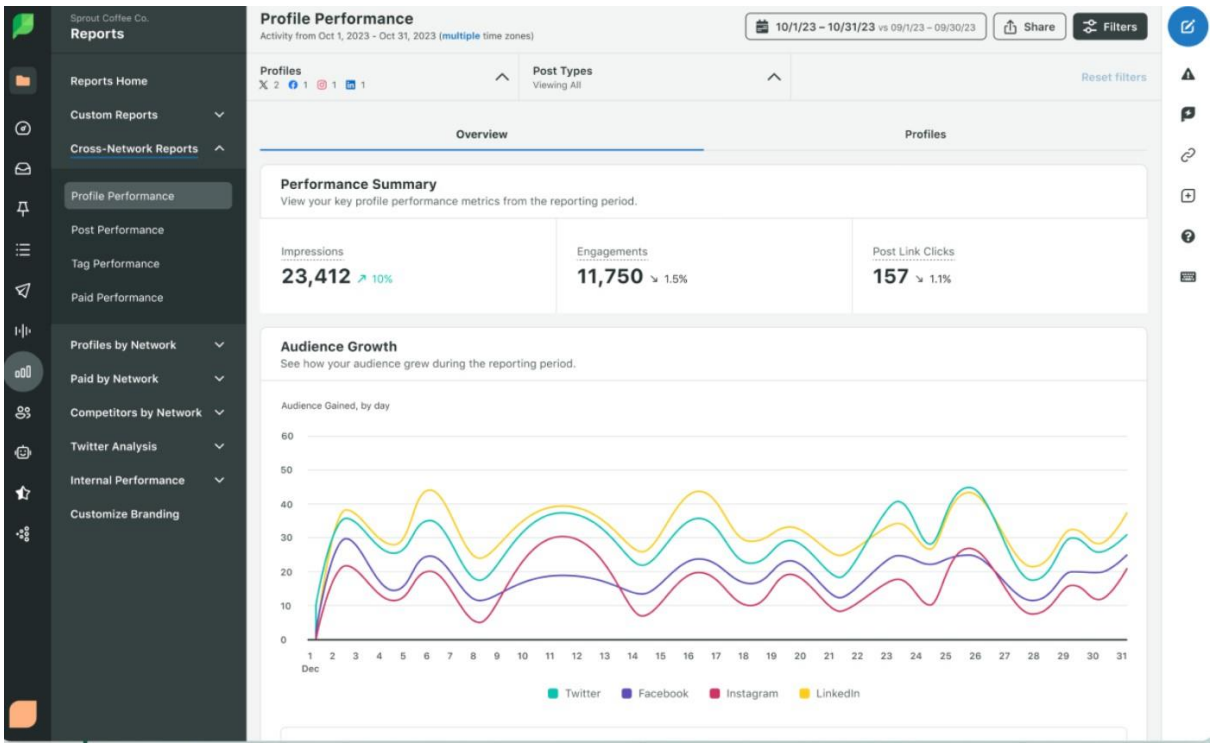

Fig. 2. Sprout Social Analysis Example

SentiStrength [26] is a tool for analysing the emotional colouring of texts, which is used mainly to assess positive and negative sentiments in short texts, such as tweets or comments on social networks. It is based on a simple approach to text analysis, using dictionaries of positive and negative words, as well as estimating the intensity of emotions. Although SentiStrength works well for English-language texts, it is not optimised for working with the Ukrainian language, which significantly reduces its effectiveness for analysing Ukrainian social networks. The main problem is that SentiStrength does not have sufficiently accurate linguistic resources to examine specific language constructions of the Ukrainian language, such as slang, surzhyk or political allusions. Also, due to the limited adaptation to the large number of meanings and expressions of the Ukrainian language, the system may incorrectly assess the emotional colouring of texts, which is critical for the analysis of public opinions in a socio-political context.

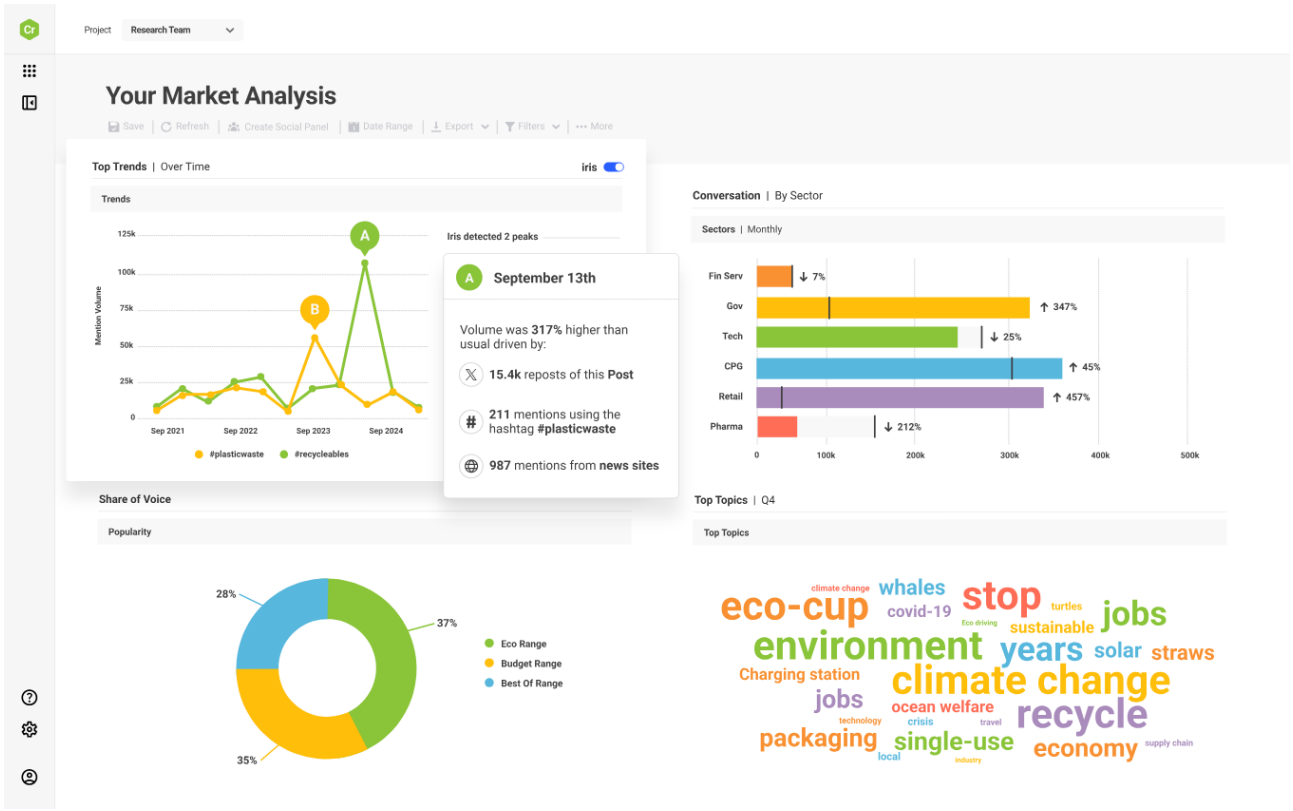

Fig. 3. Example of analysis in Brandwatch

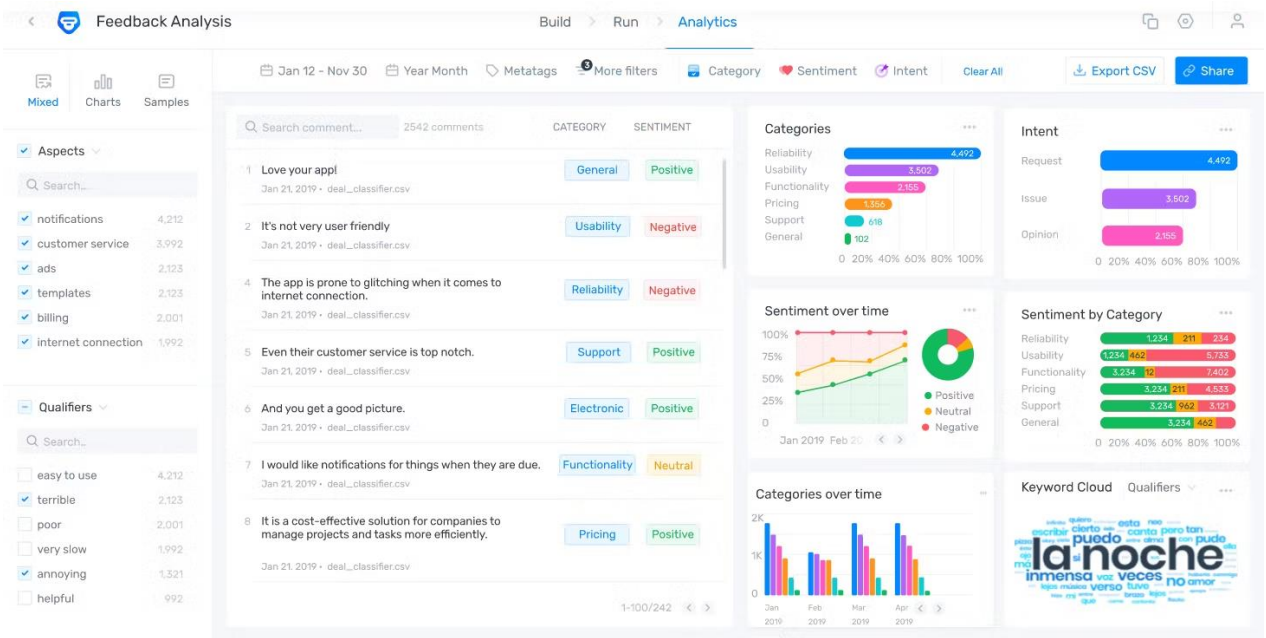

Fig. 4. Example of a working window in MonkeyLearn

TextBlob [27] is a library in Python for natural language processing that includes capabilities for sentiment analysis, grammar editing, translation, and definition of parts of speech. TextBlob uses a simple API to perform basic linguistic operations, such as determining the emotional colouring of a text. However, like most tools focused on English-language content, TextBlob does not demonstrate high accuracy when working with Ukrainian texts. The problem is that TextBlob does not support Ukrainian dictionaries for slang, surzhyk, or cultural expressions of different regions. Given this, his analysis of sentiments can be superficial or inaccurate, especially in the context of socio-political discourses, where the meaning of words can vary depending on the context or political allusions.

VADER (Valence Aware Dictionary and sEntiment Reasoner) [28] is a tool for analysing the emotional colouring of text, which is specially designed for short messages on social networks, such as tweets or comments on YouTube. VADER uses a dictionary with the designation of the tone of words and phrases and is able to effectively assess sentiment even in the context of informal statements. However, despite its popularity and effectiveness for English-language texts, VADER has significant limitations when working with the Ukrainian language. The main problem is that it does not take into account specific linguistic features specific to the Ukrainian context, such as the use of surzhyk, local slang expressions or political allusions. It causes the accuracy of sentiment analysis to be significantly reduced, especially when it comes to texts related to public figures or socio-political topics.

Sentence	Positive score	Negative score	Binary prediction
I can play chess	+1	−1	+1
I can play chess!!!	+2	−1	+1
I like to read science fictions	+2	−1	+1
I do not like to read science fictions	+1	−1	+1
I left early because the film was boring	+1	−2	−1
I hate you	+1	−4	−1
I really love you, but dislike your cold sister	+4	−3	+1

Fig. 5. SentiStrength Case Study

The analysis of modern tools shows that the problem of processing Ukrainian-language texts for mood analysis remains relevant, and the existing training samples for news and texts in English have limitations for Ukrainian analogues [29-30]. Although there are some initiatives, such as the creation of Ukrainian-language dictionaries for sentiment analysis [29] or the organization of annual workshops to find new solutions to NLP [31], they are either limited in scope and do not cover modern vocabulary, including slang, political slogans, and memes, or do not touch on the topic of socio-political discourse at all.

3. Problem Statement

Sentiment analysis in social networks is an active area of research, but most of the works focus on English-language data, while the Ukrainian-language segment remains poorly researched [5–6].

Ukrainian context

Traditional dictionary approaches in Ukrainian social networks, in particular VADER [28] and SentiStrength [26], were adapted for short texts (tweets, comments) and showed high efficiency on English-language data. At the same time, their Accuracy is sharply reduced for Ukrainian texts due to the ignorance of surzhyk, slang and political allusions [7–8]. Further work has shown that adapting dictionaries or creating new corpora for local languages significantly improves the quality of classification [9–10]. For the Ukrainian language, specialised corpora (for example, EmoBench-UA [17]) and models based on transformers, in particular Ukr-RoBERTa, appeared, which significantly surpassed classical machine learning methods (SVM, XGBoost, Naïve Bayes) in sentiment classification tasks [11–12]. Multi-platform analysis studies have shown that the genre characteristics of different networks affect the distribution of sentiments: tweets are shorter, more emotional, and more often negative, while comments on YouTube are more reasoned and tend to be neutral [13–15]. It justifies the need for a combined analysis to determine the representativeness of the results.

In Eastern European countries, there is a similar situation with low-resource languages. In Slovakia, for example, studies using SVM showed an accuracy of about 0.86 and an F1 of ≈ 0.85, outperforming other algorithms [12]. In Poland, Facebook and Twitter discourse analysis works used BERT models, but the key challenge remains the recognition of sarcasm and negative emotions. In the Russian-language segment, the emphasis is on detecting bots and manipulative content that distorts the results of the analysis [14]. For the Baltic languages (Latvian, Lithuanian), multilingual models (mBERT, XLM-R) were used, but the lack of large corpora limited their performance.

Common problems for Ukrainian and Eastern European languages are:

– low resource (lack of large open corpora and dictionaries);
– strong influence of noise in data (bots, spam, surzhik, code mixing);
– the difficulty of automatic recognition of sarcasm and political allusions [16].

At the same time, there is a tendency to switch from dictionary and classical methods to transformer models, create local corpora and expand analysis from one platform to several (Twitter + YouTube, Facebook, etc.) [17–20].

In this context, the work in question contributes to the development of the direction: it combines a multi-platform approach (Twitter and YouTube), adapts modern NLP methods to the Ukrainian-speaking environment, creates a dictionary of Internet slang and offers an interactive tool for visualising the results. It allows us to obtain more representative conclusions and overcome the limitations inherent in previous studies.

The study of sentiment analysis in the Ukrainian segment of social networks is just being formed and has a number of characteristic features:

– **Limited number of cases.** The Ukrainian language is low-resource compared to English. Traditional methods (dictionary, rule-based, e.g. VADER or SentiStrength) demonstrate low Accuracy for Ukrainian-language data due to ignoring surzhyk, jargon, spelling mistakes, and political discourse.

– **Enclosures for individual events.** Part of the work is aimed at analysing Twitter posts in connection with the war in eastern Ukraine and the full-scale invasion. Such corpora contain from hundreds of thousands to millions of tweets and are used to analyse emotional dynamics (mainly in English, but with Ukrainian tags).

– **Models for the Ukrainian language.** Comparative studies have shown that transformer models (XLM-RoBERTa, Ukr-RoBERTa, multilingual-E5) are significantly superior to classical algorithms (SVM, XGBoost, Naïve Bayes) for classifying the tonality of Ukrainian-language texts. For example, Ukr-RoBERTa reached an Accuracy of ≈ 0.92 on reviews and social media posts.

– **Specialised dictionaries.** Some works offered Ukrainian dictionaries of emotional vocabulary, but they have limited coverage and poorly reflect modern memes, political allusions, and Internet slang.

– **Multi-platform research.** There are first attempts to compare Twitter and YouTube. It is noted that tweets are shorter, more emotional, and often negative, while YouTube comments are longer and have a higher proportion of neutral or positive ratings.

In the countries of Eastern Europe (Ukraine, Poland, Slovakia, the Czech Republic, the Baltic countries), the situation is similar:

– **Poland.** Dictionary approaches and BERT models are used to analyse political discussions on Facebook and Twitter. Studies indicate difficulties in identifying sarcasm and negative emotions.

– **Slovakia.** SVM and Naïve Bayes were used for sentiment analysis; it was found that SVM on local enclosures is superior to other models (Accuracy ≈ 0.86, F1 ≈ 0.85).

– **Russian-language segment.** Much of the research focused on the analysis of Twitter messages and forums. Here, too, the problem of detecting irony, sarcasm, and bot activity is emphasised.

– **Baltic countries.** There are works on the analysis of Latvian and Lithuanian social networks, where multilingual models (mBERT, XLM-R) were used. Still, the results were lower than for high-resource languages due to the limited training data.

Generalisation:

1. Common problems:
– Low-resource languages (limited corpora, dictionaries, benchmarks).
– A large amount of noise in the data (bots, spam, mixed languages, surzhyk).
– The difficulty of automatically recognising sarcasm and political allusions.
2. Trends:
– Transition from dictionary and classical ML methods to transformers.
– Creation of specialised corps for the political and military context.
– Gradual Introduction of multi-platform analysis for better representativeness.
3. The place of the methodology under consideration. It contributes to this direction by focusing on the Ukrainian-language segment of Twitter and YouTube, creating a slang dictionary, and offering an interactive visualisation tool, which is different from many previous studies that focused only on English-language or Russian-language content.

Compared to existing systems, the developed system has a number of fundamental differences that determine its uniqueness and practical value. First of all, most of the available tools are focused on English-language content. Even those that formally support other languages generally do not provide an adequate level of accuracy when working with the Ukrainian language. In the project under development, the focus is exclusively on Ukrainian-language texts, which means taking into account the linguistic features of the Ukrainian language, the contextual features of the Ukrainian information space, and cultural and political allusions and slang, which are critical for an accurate analysis of public opinion. Another significant difference is the choice of data sources. While most systems work with either a single platform, the proposed system combines two very different types of social networks - Twitter and YouTube. It allows you to analyse both short, fast, and emotionally charged statements on Twitter, as well as more detailed comments on YouTube, usually containing arguments, irony, or evaluations of specific content. Such a multi-platform approach

provides a more complete coverage of public opinion. It reduces the risks of one-sidedness of analysis, which is often inherent in systems that work with only one data source. In addition, many ready-made tools for sentiment analysis, such as VADER or TextBlob, are based on universal English-language dictionaries and heuristics. They do not take into account the specifics of political vocabulary, slang or cultural contexts, which are key in the analysis of public figures in the Ukrainian environment.

The project is designed to teach our model of emotional tone analysis, which will be sensitive to such contexts and will be able to take into account dynamic changes in political vocabulary and current language trends in Ukrainian social networks. The main advantages of the development are Ukrainian-language orientation, the combination of two platforms, and processing flexibility. The primary, unique feature of the system under development is its Ukrainian-language orientation. Most existing systems do not support or work poorly with the Ukrainian language, and even automatic translators do not guarantee accuracy under challenging cases, such as irony, sarcasm, or political overtones. The system will be fully adapted to the Ukrainian language, taking into account surzhyk and language mixtures, political slang and neologisms, irony and sarcasm in a specific cultural context, as well as regional features of speech, which are often ignored by universal solutions. Using Twitter and YouTube at the same time creates a comprehensive effect, allowing for the analysis of both short emotional reactions and deeper comments, which broadens the analytical base and reduces research bias. It is vital for reaching a wider audience with different communication styles, as users of various platforms may have other ways of expressing opinions and varying levels of detail in their statements. The flexibility of processing is provided by the development of custom or adapted models, which allows you to take into account the context and adapt to current slang and current political events. Such adaptability means the ability to update dictionaries in accordance with language trends and quickly respond to the emergence of new terms in political discourse, which is critically important in the dynamic information environment of Ukraine.

Despite a number of advantages, the proposed approach also has its drawbacks and limitations, which should be taken into account at the design stage. Firstly, working with Ukrainian-language data is complicated by the lack of high-quality open linguistic resources related to sentiment analysis [30]. Most of the available tonality analysis dictionaries are focused on the English language, and their Ukrainian counterparts either have a limited scope or do not cover modern vocabulary, including slang, profanity, political slogans, sarcasm, and memes. It creates the need for independent development of such resources, which requires significant time and human resources. Secondly, an important challenge is the heterogeneity and non-standard nature of Ukrainian-language content in social networks. People often write with mistakes, use surzhyk, mix Ukrainian and russian, add anglicisms or transliteration. It dramatically complicates the pre-processing of texts and can negatively affect the classification quality of even the most modern machine learning models. Such features require the development of specialised text normalisation algorithms that can effectively handle such linguistic diversity. It is also worth considering the limitations associated with access to data. For example, Twitter limits the number of free API requests and also makes it difficult to access old tweets, which can limit the ability to analyse long-standing tweets. For its part, the YouTube API only allows you to receive a certain number of comments on videos, which can create problems when analysing popular content with a large number of comments. Another factor limiting the effectiveness of the analysis is the high level of irony and sarcasm on social media. Automatic recognition of such messages remains a challenge even for the most modern language models. In the context of public figures, sarcasm can significantly alter the emotional colouring of the text, and the system may incorrectly classify critical comments as positive or vice versa. In addition, there is a risk of information noise and interference by bot accounts that create a large number of messages of the same type that distort the overall picture of public opinion. This is especially true in the context of information wars and political campaigns, when artificial content generation can significantly affect the results of analysis and mislead about the real mood of society.

Despite the importance of public opinion analysis for sociology, political science, and reputation management, the Ukrainian information environment is still insufficiently covered by modern text analysis tools. Existing solutions do not allow for the complete automation of the process of studying public opinion in the Ukrainian-language segment of social networks. One of the key obstacles to the development of public opinion analysis tools in the Ukrainian segment is the lack of effective and adapted means of processing informal text content. Most of the available solutions do not provide adequate accuracy due to limited support for the Ukrainian language or ignoring lexical and stylistic features, including colloquial language, regionalisms and slang. And although there are more specialised lexical resources, there is still a shortage of them, such as dictionaries of Ukrainian-language slang or tone markers for political discourse, which are critically necessary for building accurate models of sentiment analysis [30]. As a result, the interpretation of texts dealing with public figures or socio-political topics is often simplistic or erroneous, making it difficult to reliably assess the dynamics of public sentiment.

Another problem is the lack of multi-source systems that would integrate data from several platforms at the same time. Current solutions usually work with only one platform or are reduced to an analysis of English-language tags, as a result of which the depth of analysis is lost, and the data becomes only partially representative. It is especially critical in a fragmented information space, where different social groups may predominantly use other platforms to express their opinions. The peculiarities of the Ukrainian socio-political context also complicate the situation. Modern research shows that political discourse in Ukrainian social networks has unique characteristics related to historical context, current events, and specific cultural references. Universal tools are not able to take into account these features, which leads to inaccurate interpretation of political sentiments and erroneous conclusions about the dynamics of public opinion. Thus, the central problem of the study is the need to create a comprehensive system for analysing public

opinion, which will be able to overcome both technical and linguistic barriers in the Ukrainian-language information space. Such a system should not only process texts but also deeply understand the context and convey an accurate picture of social moods, taking into account all semantic nuances specific to the Ukrainian language and culture.

4. Methodology

The study consisted of the following main stages:

1. Data collection. For the study, Ukrainian-language content from YouTube and Twitter (X) was collected in the period from *[date from the file]*. Each post (comment or tweet) was marked with metadata: time of publication, author, source, number of likes/retweets.

2. Data pre-processing: remove stop words, lemmatise Ukrainian text, and eliminate duplicates and bot content.

3. Sentiment analysis. Each document (comment/tweet) received a sentiment value S_i in the range $[-1;\ 1]$, where:

$$S_i = P_i - N_i,$$

where P_i – is the probability of a positive tonality, N_i – is the probability of a negative tonality.

4. Calculation of the average sentiment index. For the k platform (YouTube or Twitter), the average sentiment index was calculated as:

$$\overline{S_k} = \frac{1}{n_k}\sum_{i=1}^{n_k} S_i,$$

where n_k – is the number of entries from the K platform.

5. Calculating the proportion of positive/negative comments. Proportion of positive:

$$Pos_k = \frac{\sum_{i=1}^{n_k}[S_i>0]}{n_k} \times 100\%.$$

Share of negative:

$$Neg_k = \frac{\sum_{i=1}^{n_k}[S_i<0]}{n_k} \times 100\%.$$

6. Comparative analysis of platforms. To assess the difference between the mean values, a t-test was used:

$$t = \frac{\overline{S_1}-\overline{S_2}}{\sqrt{\frac{s_1^2}{n_1}+\frac{s_2^2}{n_2}}}.$$

where s_k^2 – is the sentiment variance for the k platform.

7. Visualisation. The results are presented in the form of histograms and distribution charts illustrating the prevailing sentiment for each social network. According to the calculations:

- YouTube: average sentiment index – $\overline{S_{YT}} = 0.23$, positive comments – $Pos_{YT} = 54.6\%$, negative – $Neg_{YT} = 21.3\%$.
- Twitter (X): average sentiment index – $\overline{S_{TW}} = 0.11$, positive comments – $Pos_{TW} = 48.1\%$, Negative – $Neg_{TW} = 27.5\%$.

The t-test showed a statistically significant difference in mean values ($p < 0.05$), indicating differences in the expression of emotions between platforms.

In the study, the process of training and training models for analysing public opinion from social networks took place in several stages, which can be formalised mathematically.

1. Data preparation. Set of incoming text messages $D = \{x_1, x_2, \dots, x_N\}$ with mood labels $Y = \{y_1, y_2, \dots, y_N\}$, where – $y_i \in \{-1,0,1\}$. The texts were tokenised and converted into a vector representation $X = \{\mathbf{v_1}, \mathbf{v_2}, \dots, \mathbf{v_N}\}$, $\mathbf{v_i} \in R^d$, using models such as BERT, XLM-RoBERTa, DistilBERT, etc.

2. Data breakdown. The data was divided into training (D_{train}), validation (D_{val}) and test (D_{test}) samples so that:

$$D_{train} \cup D_{val} \cup D_{test} = D, \quad D_{train} \cap D_{val} = \varnothing, \quad D_{train} \cap D_{test} = \varnothing.$$

3. Loss function. For the multiclass classification problem, the cross-entropy loss function was used:

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{ic} \log \widehat{y_{ic}},$$

where y_{ic} – is a one-hot representation of the correct class, $\widehat{y_{ic}} = \text{softmax}(z_i)_c$ – is the predicted probability of the C class, $z_i = f_\theta(\mathbf{v_i})$ – is the output of the model with the parameters θ.

4. Parameter optimization. $\mathcal{L}(\theta)$ minimization was performed by the Adam method:

$$\theta \leftarrow \theta - \eta \cdot \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}+\epsilon},$$

where $\widehat{m_t}, \widehat{v_t}$ – are the corrected moments of the gradient, η is the speed of learning.

5. Performance Evaluation. The metrics used are Accuracy Acc, Precision, Recall, F1:

$$Acc = \frac{\sum_{i=1}^{N} I(\arg\max_c \widehat{y_{ic}} = y_i)}{N}, \; F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

6. Validation and Testing. After selecting hyperparameters on D_{val}, the model was tested on D_{test} to evaluate generalizability.

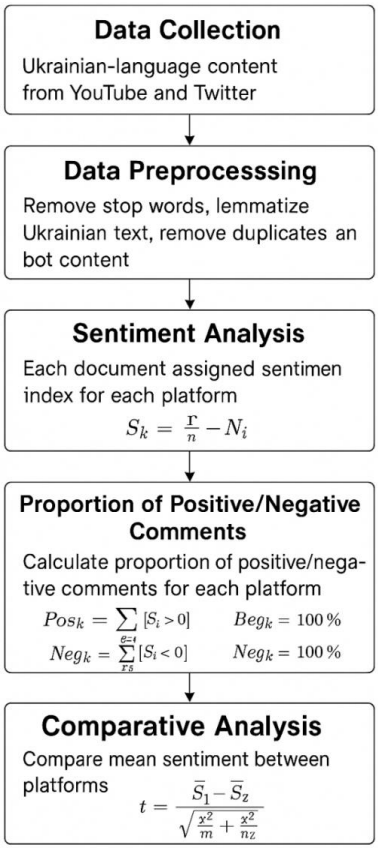

Fig. 6. Flowchart of the methodology

5. Mathematical scheme of the model training pipeline

Let the raw corpus be $\mathcal{D} = \{(t_i, m_i)\}_{i=1}^{N}$, where t_i – is the raw text (tweet/comment) and m_i – are metadata (platform, author, timestamps, likes). The annotated sentiment labels are $y_i \in \{-1,0,1\}$ (negative, neutral, positive) for labelled items.

1. Text pre-processing and normalisation. Apply deterministic cleaning functions $C(\cdot)$ and normalization $N(\cdot)$:

$$\widetilde{t_i} = N\big(C(t_i)\big).$$

Example operations: lowercasing, punctuation removal, emoji/link removal, spelling correction (Levenshtein), slang substitution.

During the pre-processing of the text, special attention was paid to the linguistic features of Ukrainian social networks: slang and jargon, surzhyk and mixed constructions, and spelling errors. A special dictionary of correspondences has been created for the most common words (for example, "зрада → поразка", "топчик → найкращий" [betrayal → defeat, topchyk → is the best]). During normalisation, such words were automatically replaced with their neutral or literary equivalents. For cases of a combination of Ukrainian and Russian (or Ukrainian and English) languages, an algorithm based on frequency dictionaries was used. If a word had a counterpart in the Ukrainian language, it was replaced by a normative form. For example, "пабєда" → "перемога", "лайкать" → "подобати" ["pabeda", → "victory", "like", → "like"]. A Levenshtein-based correction algorithm with a Ukrainian dictionary was used, which made it possible to correct the most common typographical errors and spellings (for example, "Зеленскі" → "Зеленський" ["Zelensky", → "Zelensky"]). Thanks to these steps, it was possible to significantly reduce the "noise" in the data, which is especially important for automated sentiment analysis, where even one error can change the meaning of the message.

2. Embedding (vectorisation). Text is mapped to a dense embedding using a pre-trained transformer (multilingual-e5-base):

$$\mathbf{v_i} = Embed(\widetilde{t_i}) \in R^d, d = 768.$$

Optional embedding normalisation:

$$\hat{\mathbf{v}}_\mathbf{i} = \frac{\mathbf{v_i}}{|\mathbf{v_i}|_2}.$$

Cosine similarity for two texts i, j:

$$cos(i, j) = \frac{\mathbf{v_i v_j}}{|\mathbf{v_i}|_2 |\mathbf{v_j}|_2}.$$

3. Unsupervised clustering with label propagation (hybrid annotation). Perform clustering on embeddings to accelerate/manual labelling:

3.1. Cluster embeddings with method $\mathcal{C}$ (e.g., HDBSCAN / UMAP+HDBSCAN): $\{C_1, \dots, C_K\} = \mathcal{C}(\{\mathbf{v_i}\})$.

3.2. Human expert inspects representative samples of each cluster C_k and assigns a cluster-level sentiment label $\mathrm{l}_k \in \{-1,0,1\}$.

3.3. Propagate label to all members:

$$\widetilde{y_i} = \begin{cases} \mathrm{l}_k, \text{if } i \in C_k \text{ and cluster labelled} \\ \text{(manual/exclude), otherwise} \end{cases}$$

It produces the labelled training set – $\mathcal{D}_{lab} = \{(\mathbf{v_i}, \widetilde{y_i})\}$.

4. Train/val/test split (stratified). Stratified splitting to preserve class proportions:

$$\mathcal{D}_{train}, \mathcal{D}_{val}, \mathcal{D}_{test} \leftarrow stratified\_split(\mathcal{D}_{lab};\ p_{train}, p_{val}, p_{test}, seed = r).$$

Example parameters used in the article: $N = 2989$, $p_{test} = 0.2$, $seed = 11$.

5. Models and objectives:

5.1. K-Nearest Neighbours (KNN). Prediction by majority among k-nearest neighbours (cosine / Euclidean):

$$\hat{y}(\mathbf{v}^*) = mode\big(\{y_j : j \in \mathcal{N}_k(\mathbf{v}^*)\}\big),$$

where $\mathcal{N}_k(\cdot)$ is the set of k nearest samples, with no explicit loss minimisation.

5.2. Logistic Regression (multinomial) Model $f_\theta(\mathbf{v}) = \mathrm{softmax}(W\mathbf{v} + b)$. Cross-entropy loss:

$$\mathcal{L}_\mathrm{L}(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{c\in\{-1,0,1\}} \mathbf{1}\{y_i = c\} \log \widehat{p_{i,c}}.$$

5.3. SGDClassifier (linear SVM / hinge loss variant). Hinge loss for class ccc:

$$\mathrm{l}_{hinge}(z_i, y_i) = \sum_{c \neq y_i} \max\left(0, 1 - \left(z_{y_i} - z_c\right)\right),$$

where – $z = W\mathbf{v} + b$. SGD updates weights by mini-batch / single-sample gradient steps.

5.4. XGBoost (gradient boosting) optimises an objective of the form:

$$\mathcal{L}_{\text{XGB}} = \sum_{i=1}^{n} \text{l}\left(y_i, \widehat{y_i^{(t)}}\right) + \sum_{t=1}^{T} \Omega(f_t),$$

where l – is e.g. multiclass log-loss and $\Omega(f_t) = \gamma T + \backslash tfrac12\lambda|w|^2$ regularizes tree f_t. Training uses a second-order Taylor expansion of l and greedy tree building.

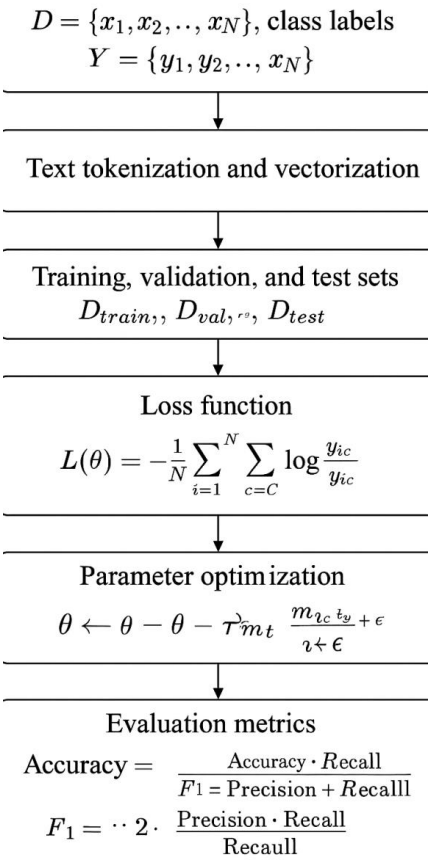

Fig. 7. Mathematical scheme of the model training pipeline

6. Optimisation and hyperparameter selection:

- Linear models (LR, SGD) trained with gradient-based optimisers (Adam/SGD). Update rule (SGD variant):

$$\theta \leftarrow \theta - \eta\nabla_{\theta\mathcal{L}}(\theta).$$

- XGBoost – gradient & hessian-based tree construction.
- Hyperparameters chosen via stratified cross-validation (k-fold) or grid/random search on validation set:

$$\text{CV-score}(\lambda) = \frac{1}{K}\sum_{j=1}^{K} \text{score}(\text{model trained } w/\ \lambda \text{ on fold } j).$$

Example hyperparameters reported:

- SGD – *alpha* $= 1e-4$, $\eta_0 = 0.01$, *loss* = *hinge*, *penalty* $= l1$.
- XGB – *learning_rate* $= 0.1$, *max_depth* $= 3$, $n\_estimators = 150$, *subsample* $= 0.8$, *eval_metric* = *merror*.

7. Training diagnostics and regularisation. Monitor loss curves $\mathcal{L}_{train}(e), \mathcal{L}_{val}(e)$ – across epochs e. Signs of overfitting: $\mathcal{L}_{train}\downarrow$ while $\mathcal{L}_{val}\uparrow$. Regularisation techniques:

- L_1/L_2 penalties – add $\lambda|\theta|_p$ to loss,
- early stopping on validation loss,
- class weights w_c – to mitigate imbalance – reweighted loss:

$$\mathcal{L}_w = -\frac{1}{n}\sum_{i=1}^{n} w_{y_i} \sum_c \mathrm{b1}\{y_i = c\} \log \widehat{p_{i,c}}.$$

8. **Evaluation metrics (per-class and aggregate).** Let TP_c, FP_c, FN_c – be counts for class c. Then:

$$Precision_c = \frac{TP_c}{TP_c+FP_c}, Recall_c = \frac{TP_c}{TP_c+FN_c}, F_{1,c} = 2 \cdot \frac{Precision_c \cdot Recall_c}{Precision_c+Recall_c}.$$

Macro- and weighted averages computed as usual. Accuracy:

$$Acc = \frac{\sum_c TP_c}{\sum_c TP_c+FP_c+FN_c}.$$

Confusion matrix M with M_{ij} entries: number of true class i predicted as j.

9. Statistical comparison of models/platforms. Use paired tests (e.g., McNemar's test) or performance comparison on the same test set. For mean sentiment comparisons across platforms, use a two-sample t-test:

$$t = \frac{\overline{S_A}-\overline{S_B}}{\sqrt{\frac{s_A^2}{n_A}+\frac{s_B^2}{n_B}}},$$

where $\bar{S}, s^2$ – are the sample mean and variance of sentiment scores.

10. Final model selection & deployment. Choose model maximizing validation F1 (or weighted-F1) with attention to:

- Generalisation gap $\Delta = score_{train} - score_{val}$,
- Inference time (KNN is slower at query time for large datasets),
- Interpretability (logistic regression is more interpretable; XGBoost provides essential features).

Persist the final model $\hat{f}$ and vectorizer:

$$\text{save}\left(\hat{f},\ \text{Embed\_config},\ \text{vocab/dicts}\right) \rightarrow \text{data/models/}.$$

Implementation notes (matching the article):

- Embeddings – multilingual-e5-base (768-dim).
- Dataset size – $N = 2989$.
- Split – stratified train/test with test_size =0.2, random_state =11.
- Tested models – KNeighborsClassifier (best final accuracy ≈ 78.8%), XGBClassifier (≈74%), LogisticRegression (≈72.2%), SGDClassifier (≈70.4%).
- Preprocessing – Stanza tokenisation, Levenshtein-based spelling correction, slang dictionary substitution.
- Annotation – cluster-based propagation, then expert correction.

Multilingual-e5-base embeddings are chosen because this model scores high in semantic search and text classification tasks in a multilingual environment. It works well with low-resource languages, including Ukrainian, in contrast to the classic TF-IDF or word2vec, which lose semantic connections in the context of political discourse.

XGBClassifier (gradient boost on decision trees) is used due to its ability to work with small and "noisy" data sets, which is typical for Ukrainian social networks. Compared to deep-tuned models (e.g., fine-tuned BERT), XGBClassifier requires fewer computing resources and provides consistent results even with limited data.

Thus, the combination of multilingual embedding and boosting made it possible to achieve a balance between Accuracy, learning speed and computational efficiency.

6. Problem Solution

To eliminate these problems, an integrated approach that includes several key components is planned to be implemented. To collect data from YouTube, it is intended to build a subsystem for interaction with the YouTube Data API v3, with the implementation of mechanisms for collecting comments from videos, and with the quota system implementation for optimal use of the API. Due to limited access to its API, the Apify tool is used to collect data from Twitter, which allows web scraping of public posts with specified parameters, including names of public figures, hashtags, and keywords. For the effective normalisation of the Ukrainian-language text, a step-by-step processing system has been applied, which includes correcting misspelt words using the Ukrainian dictionary [32], cleaning up

noise elements such as unnecessary characters, emojis and links, as well as normalising colloquial vocabulary with its dictionaries of slang and proper names. Particular attention will be paid to the processing of colloquial language. As part of the project, a dictionary of Ukrainian slang is planned to be created, which will contain common informal expressions, borrowings, distorted words, etc. Such a dictionary will allow you to more accurately interpret the meaning of messages and correctly determine their emotional tone, in particular in the context of political discussions on social networks. For text vectorisation, a ready-made multilingual embedding intfloat/multilingual-e5-base will be used [33], which will provide high-quality conversion of Ukrainian-language texts into numerical representations, while maintaining semantic meaning even for specific language constructions. Data labelling for model training will be carried out by hand, taking into account the context of Ukrainian socio-political discourse.

Within the framework of the project, it is planned to create a machine learning model for analysing the tonality of Ukrainian-language texts, which will be taught in the paradigm of supervised learning using a corpus of texts with predefined tonality classes. The architecture of the model will be based on a classifier using vector representations of text from multilingual-e5-base. It is planned to experiment with various machine learning algorithms, including the nearest neighbour method (KNN), gradient stochastic descent (SGDClassifier), and the gradient boosting algorithm (XGBClassifier), followed by the selection of the most efficient approach based on accuracy metrics.

To visualise the results and analytics, a system for building interactive dashboards with graphs of public opinion is planned, with the ability to compare results between public figures and platforms for publishing comments. The system will provide the ability to filter by various criteria and obtain detailed statistics on the distribution of sentiment. The interface will allow users to select a specific public figure or social network for analysis, after which the system will display the distribution of sentiment in the form of bar charts. For example, to analyse a specific person, the number of positive, negative and neutral messages will be shown, and to compare platforms, the distribution of sentiment separately for YouTube and Twitter will be shown, which will reveal the peculiarities of perception on different social networks. The analysis showed that the existing solutions have critical shortcomings in the context of processing Ukrainian-language content, which justifies the need to develop a specialised system. The proposed approach is characterised by a Ukrainian-language orientation, the involvement of various platforms for collecting information, and contextual adaptability, which will provide significantly higher accuracy of public opinion analysis in the Ukrainian information space compared to existing solutions.

The goal of this project is to create an information system for automated collection, processing, analysis and visualisation of data from social platforms (Twitter and YouTube) in order to determine public opinion about individual public figures based on the study of the emotional tone of text messages. As part of the development, the following tasks are set:

- Implement modules for collecting data from external sources (Twitter API, YouTube).
- To carry out preliminary processing of the Ukrainian-language text, taking into account its linguistic and stylistic features;
- Build or adapt a machine learning model for analysing the sentiment of texts.
- To provide visualisation of results with the possibility of comparative analysis by platform, person, and period.

Data collection from various social platforms is implemented using separate subsystems, each of which is optimised for the peculiarities of access to content on a particular platform. To ensure the balance of the dataset, an equal number of comments for each public figure is planned to be collected. YouTube: To get comment texts under the video, we have developed our parser based on the YouTube Data API. It allows you to filter by channel names, keywords, names of public figures, and publication dates. The youtube_comments.py module provides comment collection with the ability to filter and structure data in detail. Twitter: Given the restrictions on access to the Twitter API and the requirements for a paid subscription, the ready-made Apify service is used to collect tweets, which provides web scraping of public content according to specified parameters (names of public figures, hashtags, keywords, etc.). It allows you to flexibly configure collection scenarios without the need for direct API access.

The data_aggregation.py module integrates data from different sources, their initial processing, and their structuring. The names_data.py module is responsible for working with the names of public figures and systematising them for further analysis.

A multicomponent processing system is used to normalise Ukrainian-language text effectively. The preprocessing_text.py module performs basic cleaning of noise elements (extra characters, emojis, links), and the correction.py module is responsible for correcting common errors and normalising text. The preprocessing_data.py module provides general coordination of processing processes and preparation of data for modelling. Particular attention is paid to the processing of colloquial vocabulary. It is planned to create a custom dictionary of Ukrainian slang, which will contain common informal expressions, borrowings, distorted words, etc., as well as a glossary of proper meanings. It will allow you to more accurately interpret the meaning of messages and correctly determine their emotional tone.

It is planned to create our machine learning model in the teacher-led learning paradigm, which classifies texts into three emotional categories: positive, negative and neutral. The module is train.py, which is responsible for training the model, and predict.py is accountable for applying the trained model to new data.

The interface will allow users to select a specific public figure or social network for analysis, after which the system will display the distribution of sentiment in the form of bar charts. For example, to analyse a specific person, the number of positive, negative and neutral messages will be shown, and to compare platforms, the distribution of sentiment separately for YouTube and Twitter will be shown, which will reveal the peculiarities of perception on different social networks. The project is called the Information System for Analysing Public Opinion Based on Text Content from Twitter and YouTube. Purpose of the system:

- automated collection of text content from YouTube and Twitter;
- aggregation and systematisation of data on public figures;
- text pre-processing, including normalisation, lemmatisation, and purification;
- analysis of the emotional tonality of Ukrainian-language messages;
- visualisation of the results in a convenient form for further analysis.

Functional requirements of the main modules of the system:

1. Data Collection Module:
- Collect comments from YouTube using the YouTube API;
- Collecting tweets through the Apify platform (web scraping);
- Aggregation of data from various sources;
- Saving the collected data in a structured form (CSV);

2. Word Processing Module:
- Cleaning the text from noise symbols (links, emojis, etc.);
- Correction of grammatical errors in the text;
- Separation of moulded words;
- Application of a custom dictionary for the interpretation of slang:

3. Sentiment analysis module;
- Training a machine learning model to classify text into emotional categories.
- The model is applied to analyse the collected content.

4. Visualisation Module:
- Construction of bar diagrams of the distribution of emotional tonality;
- Comparison of public opinion on different public figures and platforms.

Non-functional requirements are that the system must have a modular structure:

- Scalability (adding new sources or new public figures);
- Use of open source software;
- Convenient directory structure for storing raw and processed data.

Main restrictions:
- Twitter data is not accessed through APIs, but through Apify (web scraping).
- The analysis is carried out mainly on data from YouTube.
- The analysis is carried out exclusively on Ukrainian-language texts.

Technologies and tools:

- Programming languages: Python 3.11.9;
- Libraries: Pandas, NumPy, Scikit-learn, Matplotlib, Seaborn, Stanza, XGBoost, sentence_transformers, tqdm, joblib, umap, hdbscan;
- Data collection: YouTube API, Apify;
- Rendering: Matplotlib, Seaborn, Streamlit.

Development environment:

- Jupyter Notebook is used at the stage of research, data collection and analysis (EDA), hypothesis testing, building and testing of machine learning models, both for clustering and labelling data, and for finding the best model of supervised learning. Allows you to quickly visualise the results and adapt the code.
- VS Code - used to write modular code in the form of .py files, implement system architecture, collect data, prepare processing pipelines, save and deploy models, and create a web interface through Streamlit.

The system is implemented according to a modular approach, where src/data_collection/: data collection and aggregation, src/preprocessing/: preparation, correction and cleaning of text, src/modeling/: training and application of data models/: storage of data and models, notebooks/: experimental research and visualisation. Expected results:

- Structured data from YouTube and Twitter.
- Trained a model for sentiment classification.
- Visualisation of public opinion in graphic form through a web interface.
- A ready-made system that can be scaled or adapted to other sources.

The information system being developed consists of several main functional subsystems, which together provide a complete cycle of work with data, from collection to visualisation of results. The data collection subsystem is responsible for obtaining information from external sources. The project provides for the use of two leading platforms: Twitter and YouTube. To collect data from YouTube, we have developed our module that interacts with the YouTube Data API and filters content by names of public figures, keywords and channel names. Data collection from Twitter is carried out using the Apify service, which provides access to tweets with the necessary filters. The data preprocessing subsystem prepares text information for further analysis. It includes a preprocessing_text.py module for basic text cleaning from noise (extra characters, links, emojis), a correction.py module for error correction, separation of moulded words and normalisation of slang words, and a preprocessing_data.py module for coordinating all processing processes and preparing data for model training. For the correct interpretation of colloquial vocabulary and slang, it is planned to use the developed slang dictionary and proper names. The modelling subsystem implements machine learning algorithms to analyse the emotional tonality of texts. The train.py module is responsible for creating and teaching a model for classifying texts into three categories: positive, negative and neutral. The predict.py module applies the trained model to new data and generates predictions. The visualisation and user interface subsystem is designed to display the results of the analysis in the form of interactive graphs that allow you to compare public opinion on different public figures, as well as track changes in the time dimension. The Streamlit library is used to implement the interface, which provides a simple and user-friendly web interface.

Administrator – responsible for system configuration, control and launch of data collection and update processes, training and validation of machine learning models, as well as updating the interface and database. The administrator has full access to all system technical functions and ensures its stable operation. The user has access to the system's web interface, which is implemented using the Streamlit library. Can view the results of public opinion analysis in the form of bar charts, select specific public figures for analysis, and compare the distribution of sentiment between platforms (Twitter, YouTube). The user does not have access to internal data collection or processing processes. Interconnections of components:

- Data collection modules (youtube_comments.py, data_aggregation.py) receive raw text data from external platforms and structure it for storage.
- The preprocessing subsystem (preprocessing_text.py, correction.py, preprocessing_data.py) takes raw data, cleans it, and normalises it so that the models work correctly.
- Modelling modules (train.py, predict.py) receive the prepared data and classify texts by tone.
- The results are transmitted to the visualisation subsystem, where interactive graphs and reports are generated.
- Users have access to visualisations and the ability to filter results through the Streamlit interface.

Information flows:

- Data collection stream - data from Twitter and YouTube is collected by the appropriate modules, aggregated and systematised, and then stored raw in the data/raw/ directory.
- Pre-processing flow - raw data undergoes cleaning, error correction, and normalisation, taking into account the specifics of the language, as well as the use of slang dictionaries and proper names to improve the quality of texts. The processed data is stored in data/processed/.
- Analysis flow - the cleaned data is fed to the input of the model, which performs the classification and determines the emotional tone of each text message. Trained models are stored in data/models/.
- Visualisation flow - classification results are aggregated, appropriate metrics and bar charts are generated, and the results are displayed to users through a web interface with the ability to interactively select analysis parameters.

The architecture of the system is implemented on a modular principle with a clear division into functional components that correspond to the main stages of data processing and analysis from collection to visualisation of results.

- src/data_collection/ – module for collecting and aggregating data from different platforms;
- youtube_comments.py – the main module for collecting comments from YouTube through the YouTube Data API, with the ability to filter by channels, keywords and public figures;
- data_aggregation.py – aggregation and merging of data from YouTube and Twitter, ensuring their structured storage;

- names_data.py – creation of a dictionary with different cases of the names of public figures, for their further normalisation in the text;
- src/preprocessing/ – text preprocessing module;
- preprocessing_text.py – basic text cleaning from noise symbols, emojis, links, extra characters;
- correction.py – correction of grammatical errors, the use of prepared dictionaries to correct typical spelling errors, bringing slang words to their standard substitutes and separating moulded words;
- preprocessing_data.py – coordination of all processing processes, preparation of data for modelling src/modelling/ – machine learning module;
- train.py – training a machine learning model to classify emotional tonality, preparation of the training sample, cross-validation and preservation of the model;
- predict.py – application of the trained model to new data, prediction of the sentiment category (positive, negative, neutral), preservation of results;
- data/ – data storage module;
- data/raw/youtube_data/ – raw data received from YouTube and Twitter;
- data/processed/ – texts after cleaning and processing;
- data/models/ – stored trained machine learning models;
- data/dicts/ - dictionaries of Ukrainian words, names of public people, slang;
- notebooks/ – module with experiments;
- eda_data.ipynb – data research, clustering experiments for labelling data, as well as testing models to determine the best;
- YouTube_Comments_Advanced.ipynb – experimental data collection from YouTube.

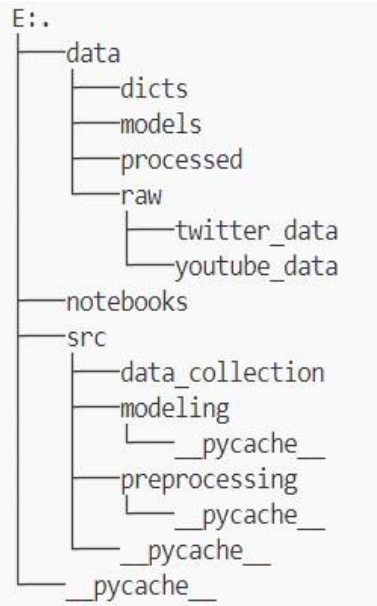

Fig. 8. Project structure

There are two types of users: a regular user and an administrator. An ordinary user can only view visualisations (graphs) and filter data according to which public figures to display or from which platform comments about them were collected. The administrator (developer) can start collecting and processing texts, train the model, and then run predictions. In the future, the administrator can update the slang dictionary if necessary. Based on the processes available to the user, viewing visualisations, and filtering data (by public person and/or by platform), the main processes available to the administrator are starting data collection, starting text processing, training the model, running predictions, and updating slang dictionaries. Main scenarios:

- filtering data and viewing visualisations by them;
- data collection;
- text processing;
- model training;
- conducting a prediction;
- Slang dictionary updates.

An activity diagram shows processes in the form of activity flows. For example, an activity diagram for public opinion prediction shows the path from collecting people's sayings on the Internet and processing the assembled corpus to training the model on the collected data and running predictions on it. The main processes of the pipeline:

1. Start data collection.
2. Start processing the assembled case.
3. If the Data is processed for training, then train the model and save the trained model.
4. If the data is processed for prediction, then run the prediction.

This diagram (Fig. 9) shows the key classes of the system, such as "Opinion Analysis", which is the final class with which both the user and the administrator can interact, as well as the "Trained Model" and "Prediction" classes, which receive data after passing the "Pre-processor" class, where the text that is obtained from the "Data Aggregator" and "Data Collection" classes is processed. This diagram (Fig. 10) depicts the physical components of the system, such as the client interface, backend, database, and third-party APIs (such as a map service or TMDB API). It demonstrates how components are related to each other and what data is transferred between them.

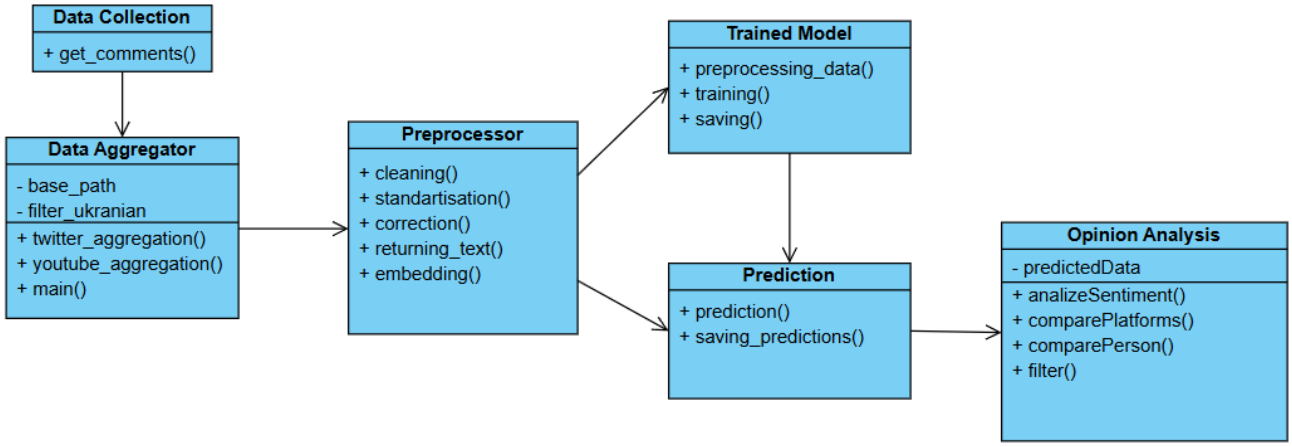

Fig. 9. Class Diagram

In the process of developing a system for analysing public opinion in social networks, it was necessary to choose the optimal methods and tools for each stage of data processing. Decisions were made on the basis of a comprehensive analysis of available alternatives, taking into account the specifics of the Ukrainian text and the limitations of computing resources. Choosing a programming language was one of the easiest decisions in system development. Python ranked first in the June 2025 TIOBE Index with a rating of 25.87% [34], confirming its leadership in data analytics. Compared to R, Python shows better results in terms of the speed of project implementation and has a larger arsenal of libraries for machine learning [35]. Java, while providing higher execution performance, requires significantly more development time and does not have such a large selection of specialised tools for natural language processing [36-37].

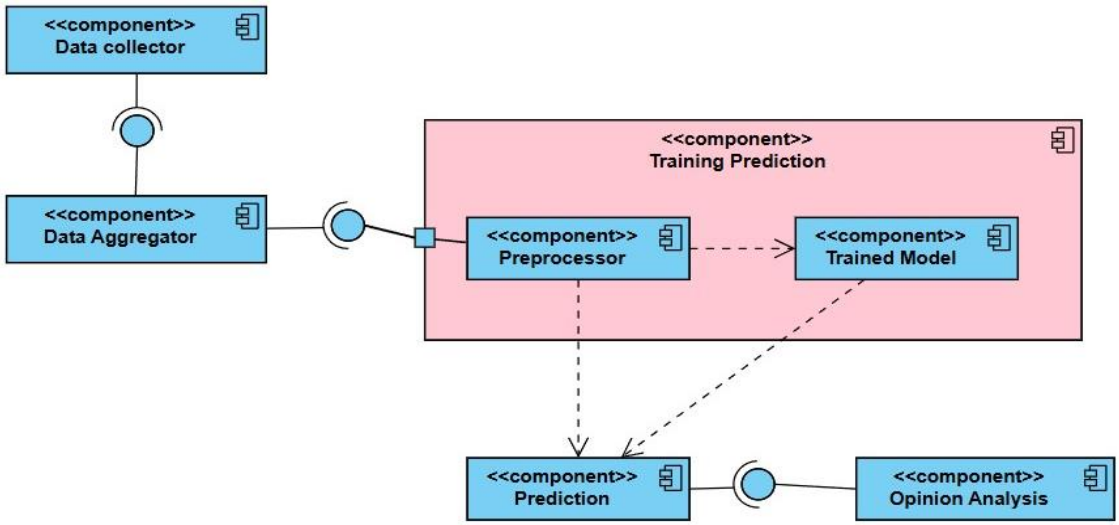

Fig. 10. Component Diagram

Table 2. Comparison of programming languages

Criterion	Python	R	Java
Development speed	High	Average	Low
Support for ML libraries	Excellent	Dobra	Average
Performance	Average	Low	High

An additional advantage of Python is its interpreted nature, which allows you to quickly test hypotheses and experiment with different approaches. The PyPI package ecosystem contains more than 400,000 packages, of which a significant portion is dedicated to data analysis and machine learning. Given the need for rapid prototyping and experimentation with different approaches, Python 3.11.9 turned out to be the best choice for our project.

Data collection required a special approach due to the limitations of official social media APIs. For Twitter, the Apify platform was chosen, which bypasses the restrictions on the number of requests and provides convenient access to historical data. Compared to using the official Twitter API, Apify allows you to get significantly larger amounts of data without the need for complex authentication setup and limit monitoring [38-39].

Table 3. Comparison of Twitter Data Collection Methods

Setting	Apify	Twitter API v2	Scrapy + BeautifulSoup
Request restrictions	Minimum	15000 posts/month	Depends on IP
Cost	$39/month (free credits available)	$200/month	Free
Complexity of setup	Low	High	Very high
Stability	High	High	Low
Risk of blocking	Low	Missing	Very high

Alternative solutions, such as writing scrapers yourself with eautifulSoup or Scrapy, need to be constantly updated due to changes in the structure of social media web pages and can lead to IP blocking. Additionally, such solutions do not guarantee the receipt of complete metadata of publications, which is critical for analysing the context and dynamics of public opinion. For YouTube, it was decided to use the standard YouTube Data API, as it provides stable access to comments with the necessary metadata. The YouTube API allows you to receive up to 10,000 requests per day for free, which is enough for most analytics tasks. The API provides structured data that can be stored in CSV format, including information about the author of the comment, the time of publication, the number of likes and replies.

Text preprocessing is a critical step, especially for Ukrainian-language texts from social networks. When choosing a library for linguistic processing, NLTK, spaCy and Stanza were compared. The NLTK library was almost immediately rejected because it does not support the Ukrainian language. Stanza was chosen because of its full support for the Ukrainian language and high accuracy of morphological analysis, which is critically vital for high-quality processing of texts from social networks.

A separate challenge was the need to correct spelling mistakes and process slang characteristic of social networks. A multi-level processing system has been developed.

The leading dictionary of valid words was taken from the extensive electronic dictionary of the Ukrainian language (VESUM) [35, 37], which contains more than 3.5 million word forms. Two other dictionaries - slang words and their literary counterparts with 108 pairs, as well as a dictionary with 53 names of public figures, and various cases of words mentioned in socio-political discourse - were created independently.

A multilevel algorithm based on the Levenshtein metric with optimisation through caching has been developed. The system works according to the following algorithm.

- Check the leading dictionary. First, the presence of the token in the dictionary of valid words and the dictionary of proper names is checked.
- If a word is not in the leading dictionary, the system checks it in a slang dictionary with 108 pairs. When a match is found, the slang word is replaced with a literary equivalent.
- A recursive partitioning algorithm is used to analyse tokens that can be compound words. The system tries to find the longest valid prefix and recursively processes the remainder.
- For unknown words, the Levenshtein distance to all words in the dictionary is calculated with preliminary filtering by length (the difference is no more than max_distance characters). First, a conservative threshold is applied (distance ≤ 2), then a more liberal one (distance ≤ 4).
- The @lru_cache(maxsize=1000) decorator is used to save the results of the most frequent queries, which significantly speeds up the processing of duplicate words.

The correct_word() function implements an optimised search for the nearest word by filtering candidates by length and calculating the minimum Levenshtein distance. The process_tokens() function coordinates the entire process of processing a sequence of tokens, applying the described steps in a defined order. The system uses several optimisation methods to ensure that large amounts of text are processed quickly. Caching through LRU_cache for a thousand words reduces the number of calculations for duplicate words. Initially, 10,000 was taken, but this took up a lot of virtual memory, so it was decided to reduce it to 1000. Pre-filtering candidates by length in the function correct_word()

dramatically reduces the number of Levenstein distance calculations. Using frozenset for dictionaries provides a quick O(1) lookup to check for word presence.

Text vectorisation required a careful choice between traditional methods and modern transformer models. Primary experiments with TF-IDF and Word2Vec showed low classification accuracy due to the limited ability of these methods to take into account context and semantic relationships between words. TF-IDF, being a statistical method, does not take into account the order of words and their semantic proximity, which is especially critical for sentiment analysis, where context plays a decisive role. Word2Vec, together with TF-IDF, showed better results due to their ability to form semantically meaningful vector representations of words; however, these methods still have limitations in considering context at the sentence level. Instead, the use of multilingual-e5-base provided significantly better results due to its ability to form contextual vector representations. This model is optimised explicitly for multilingual tasks and demonstrates high efficiency for the Ukrainian language, which is confirmed by experimental results. The model is based on the BERT architecture and has been trained on a massive corpus of multilingual texts, including Ukrainian content.

The choice of machine learning algorithms was based on the need to balance between accuracy and processing speed. Several approaches to solving the problem of tonality classification are considered, each of which has its advantages and limitations.

- Stochastic Gradient Descent (SGD) is considered a basic algorithm due to its efficiency in working with large amounts of text data and supporting online learning. This algorithm updates the model weights based on each object, making it particularly suitable for working with social media streaming data. An additional advantage is the ability to use various loss functions, including hinge loss for SVM and log loss for logistic regression.
- Logistic Regression is considered a basic model for comparing results. Despite being simple, logistic regression shows consistent results and is easily interpreted, which is vital for understanding the factors that influence sentiment classification.
- K-Nearest Neighbours can be used to analyse local dependencies in data. KNN is helpful in detecting anomalies and analysing complex cases where linear models can give false results, although it has limitations on the prediction rate.
- XGBClassifier, gradient boosting, is considered to evaluate the potential of more complex ensemble methods. This algorithm usually shows high accuracy, but requires more time to learn and predict.

The quality assessment system includes a comprehensive set of metrics for a thorough analysis of the model's performance. The main classification metrics are accuracy, precision, and recall to evaluate the effectiveness of each key class. The F1 score is a harmonic average of precision and recall. Additionally, vectorisation quality metrics are monitored through cosine similarity between semantically similar texts and a confusion matrix for detailed analysis of classification errors. Runtime metrics are also essential to assess the scalability of the system. To track the dynamics of model learning, a loss function graph is used, which allows you to monitor the convergence process and identify signs of overtraining or undertraining.

The visualisation and interface tools were chosen taking into account the speed of development and functionality. Several alternatives have been considered for creating an effective data analysis interface. The main task was to create interactive dashboards to visualise the results of public opinion analysis. In this regard, Gradio was not a good fit, as this library is mainly for demonstrating ML models and creating simple I/O interfaces. Dash was also rejected due to the excessive complexity of the setup for the relatively simple rendering tasks of the project.

- Matplotlib and Seaborn provide the necessary capabilities for creating statistical graphs and integrating them with Pandas without additional dependencies. Matplotlib allows you to make basic graphs quickly and efficiently, while Seaborn adds statistical capabilities and improved visual style. The combination of these libraries covers all the needs of visualising the results of sentiment analysis.
- For the demo interface, Streamlit was chosen, which allows you to quickly create web applications without in-depth knowledge of web development. Compared to Dash or Gradio, Streamlit combines ease of use with enough flexibility to create an intuitive sentiment analysis interface. Streamlit automatically processes the state of the application and ensures that the interface is responsive when the parameters change. The interface includes interactive elements for configuring analysis parameters and setting filters by public persons to be investigated or the platform from which the data was taken.

Visual Studio Code was chosen for system development as the main integrated development environment, providing an optimal balance between functionality and performance. A key advantage of this solution is the built-in support for Jupyter Notebook, which allows you to combine structured code development with interactive data analysis in one environment. VS Code provides a wide range of capabilities for developing Python applications, including intelligent code completion, an integrated debugging system, and support for Python virtual environments. Extensions for Python provide full language support with syntactic highlighting, error checking, and code refactoring. Jupyter Notebook is used as an additional tool for experimental data analysis, hypothesis testing, and visualisation reporting.

This technology allows you to quickly test ideas, visualise intermediate research results, and document the development process. Especially valuable is the ability to save the results of cell execution and their further use for analysis. Combining VS Code with Jupyter Notebook creates an efficient development cycle where you can conduct step-by-step testing of algorithms in notebook files, and then transfer the tested code to structured Python modules. This approach provides both flexibility for experimentation and support for high-quality software product architecture. The conducted comprehensive analysis made it possible to form an optimal technology stack for the public opinion analysis system, which provides high efficiency at all stages of data processing. The selected solutions demonstrate a synergistic effect when used together and create a reliable basis for solving complex problems of sentiment analysis of the Ukrainian text. The use of Python, with its rich ecosystem of specialised libraries, provides speed of development and ample room for experimentation. The combination of Apify to collect data from Twitter and the official YouTube API creates a stable and efficient channel for extracting data from major social platforms. Stanza for linguistic processing and multilingual-e5-base for vectorisation form a potent tandem for processing Ukrainian-language text, given its morphological complexity and specifics of informal communication in social networks.

The development of sentiment analysis software is a complex task that requires the integration of natural language processing and machine learning methods. Taking into account the specifics of the Ukrainian language and the peculiarities of social network texts, the system should effectively handle informal communication style, slang, spelling mistakes and emotionally colored statements. The main challenge in the development of such a system is the need to take into account the linguistic features of the Ukrainian language, in particular the rich morphological structure, the variability of spelling words in an informal environment and the availability of a specific vocabulary of political discourse. Therefore, the system's architecture was designed taking into account these factors and built on the principles of modularity to ensure flexibility and the possibility of further development.

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-layer model, where each layer performs specific data processing functions. The lower level is responsible for the collection and primary processing of raw data, the middle level carries out data preprocessing, linguistic processing, and vectorisation, and the upper level implements machine learning and classification. The structure of the project is organized as follows: the 'data' folder contains all project data, including raw data from social networks in the 'twitter_data' and 'youtube_data' subfolders, processed data in 'processed', trained models in 'models', and auxiliary dictionaries such as a dictionary of valid words, proper names, or slang in 'dicts'. The 'src' directory contains the main program code, distributed by functional modules. 'data_collection' for data collection, 'preprocessing' for text processing, 'modelling' for training models and predicting results. Additionally, the 'notebooks' directory contains Jupyter notebooks for experimental research and prototype algorithms.

Creating a high-quality dataset for analysing the sentiments of Ukrainian-language texts required an integrated approach to data collection, filtering, and markup. The process began by collecting comments from two primary social media sources representing different communication styles and audiences. The first source is made up of comments and responses from Twitter users regarding public figures, characterised by brevity of statements, high emotional saturation and frequent use of informal vocabulary. The second source is represented by YouTube comments under videos of interviews and speeches by public figures, which are usually more detailed, reasoned and contain more detailed justifications for users' positions. A critical step was the development of a hybrid approach to markup of collected texts, combining automatic clustering with expert manual markup to ensure an optimal balance between process efficiency and quality of results. Traditional, fully manual markup of large amounts of textual data is a highly time-consuming process that can lead to inconsistency due to expert fatigue and subjectivity of assessments. On the other hand, fully automatic markup methods often do not take into account the subtleties of language and contextual features, which are especially critical for analysing sentiments in informal social media texts.

The developed approach began with the automatic clustering of the entire assembled corpus of texts using unsupervised machine learning algorithms. For clustering, vector representations of texts obtained using the multilingual-e5-base transformer were used, which provided a semantically significant grouping of comments with similar content and tonality. The clustering algorithm automatically detected natural groups in the data, combining texts with similar characteristics. After the clustering was completed, the analysis of the resulting clusters was carried out to understand their semantic content and tonal characteristics. Each cluster was investigated through a selective reading of representative examples, which allowed the identification of the dominant themes and emotional colourings of the group. Based on this analysis, each cluster was assigned a sentiment label: negative, neutral or positive, in the form of -1, 0 and 1, which automatically spread to all texts in the corresponding group. This approach has provided several significant advantages compared to traditional marking methods. Firstly, the time required for the markup process was significantly reduced, since instead of individual processing of each text, it was enough to analyse a limited number of clusters. Secondly, the consistency of markup increased because all texts with similar characteristics automatically received the same marks, which minimised subjectivity and inconsistency of assessments.

The effectiveness of the developed approach applies to the entire pipeline of sentiment analysis, which includes:

– data collection (YouTube + Twitter),
– pre-treatment (noise cleaning, normalisation, spelling correction, work with slang/surzhyk),
– vectorisation using multilingual embeddings (multilingual-e5),
– classification using XGBClassifier,

– evaluation of results by Accuracy and F1 metrics.

Thus, the approach is seen as a holistic system, not just a classification model. At the same time, it is worth clarifying that the central part of the performance obtained is explained precisely by the quality of the mood model (XGBClassifier + multilingual embeddings).

7. Experiments

The central module of the system is responsible for coordinating the entire process of training the model from the initial stage of data loading to the final saving of the results. The module implements a comprehensive input validation system, which includes checking the integrity of CSV files, the presence of required columns, and the correctness of class labels. Particular attention is paid to the stratified division of data with the selected random_state into training and test samples, which ensures the preservation of a proportional representation of all classes of sentiments in the test and training samples, which can be reproduced for training predictions.

```
def main():
    data_path = get_resource_path("data/raw/ukrainian_social_data.csv")         try:
        data = pd.read_csv(data_path)
        print(f"Loaded {len(data)} entries")                  train, test =
train_test_split(data, test_size=0.2, random_state=11, stratify=data['sentiment'])
train = train.dropna(subset=['text', 'sentiment'])         model_trainer =
Model(train)         model_trainer.saving()
        print("Model trained and saved!")
```

The module's error handling system includes detailed logging of all stages of execution with the ability to track problem areas of the code. A flexible system for configuring project resource paths has been implemented, which makes it easy to adapt the system to different working environments. The module also provides automatic creation of the necessary directories to save intermediate results and final models.

```
    except FileNotFoundError as e:
        print(f"Error: File not found - {e}")     except
pd.errors.EmptyDataError:
        print("Error: Data file is empty or corrupted")     except KeyError as e:
        print(f"Error: Missing required column - {e}")     except Exception as e:
print(f"Unexpected error: {e}")         import traceback traceback.print_exc()
```

The text correction module (correction.py) represents one of the most critical components of the system, since the quality of processing the informal text of social networks directly affects the accuracy of sentiment classification. The correction system is built on the basis of three primary dictionaries: a dictionary of valid words of the Ukrainian language, which is loaded from the 'words_spell.txt' file, a personally formed dictionary of slang in JSON format with pairs of slang expressions and their dictionary equivalents, and a dictionary with various variations of the spelling of proper names of public figures. The word correction algorithm is based on calculating the Levenshtein distance between an erroneous word and words from a valid dictionary. The 'correct_word' function searches for the word closest in terms of editorial distance with a maximum distance limit of three characters, which provides a balance between the accuracy of correction and the performance of the algorithm. To process moulded words, the 'split_word' function is implemented, which uses dynamic programming to find the optimal splitting of a sequence of characters into valid words. Particular attention is paid to the processing of slang through the 'slang_correction_token' function, which replaces informal expressions with their literary equivalents. The primary function 'process_tokens' combines all stages of correction and ensures sequential processing of tokenised text, taking into account the context and morphological features of the Ukrainian language.

```
def process_tokens(tokens):
    result = []     for token in tokens:         if token in main_dict:
result.append(token)             continue          slang_corrected = slang_correction_token(token,
slang_dict_map, slang_keys)         if slang_corrected:
            result.extend(slang_corrected)             continue
split = split_word(token, main_dict)         if split:
result.extend(split)             continue          corrected =
correct_word(token, main_dict, 2)         if corrected in main_dict:
result.append(corrected)             continue
         corrected = correct_word(token, main_dict, 4)         if
corrected:
            result.append(corrected)         else:
            result.append(token)
    return result
```

A specialised module for processing Ukrainian-language texts of social networks (preprocessing_text.py) is implemented in the form of the 'PreprocessorText' class, which covers all stages of transforming the source text into structured data suitable for further work, namely training a machine learning model. The processing process is

organised into three main phases, each of which performs specific functions of cleaning and standardising the text. The cleaning stage includes the removal of user mentions in the '@username' format, which is typical for social networks and does not carry a semantic load for sentiment analysis. The system also cleans punctuation and special characters, taking into account the peculiarities of Ukrainian typography, removes link URLs, and normalises spaces to eliminate redundant formatting characters.

```
def cleaning(self):         self.text = self.text.str.replace(r'@\w+', '', regex=True)   # delete
usernames         self.text = self.text.str.replace(r"[^\w\s]", '', regex=True)         self.text
= self.text.str.replace(r'[\r\n]+', ' ', regex=True)         self.text =
self.text.str.replace(r'\b[a-zA-Z]+\b', '', regex=True)         self.text =
self.text.str.replace(r'\d+', '', regex=True)         self.text =
self.text.str.replace(r"http\S+", "", regex=True)         self.text = self.text.str.replace(r'
+', ' ', regex=True)         self.text = self.text.str.strip()         self.text =
self.text.str.lower()         return self.text
```

The standardisation stage is based on the use of the Stanza library, specially configured for the Ukrainian language, which provides tokenisation taking into account the morphological features of the language. To optimise memory usage, parallel batch processing is implemented, which allows you to efficiently process large amounts of text without exceeding system resource limits. At this stage, word correction is also applied through the previously described function 'process_tokens'.

```
def standartization(self, batch_size=None, n_jobs=None, memory_limit=85):
        token_pipeline = stanza.Pipeline(lang='uk', processors='tokenize')         self.text =
self.text.apply(lambda x: [word.text for word in token_pipeline(x).sentences[0].words])
total_rows = len(self.text)                 if batch_size is None:             if total_rows <
100:
                batch_size = total_rows             elif
total_rows < 1000:                 batch_size = 100
else:
                batch_size = min(200, total_rows  10)                 if
n_jobs is None:             if total_rows < 100:
                n_jobs = 2             elif
total_rows < 1000:
                n_jobs = 4             else:
                n_jobs = 6                 all_processed = []
for i in tqdm(range(0, total_rows, batch_size), desc="Processing batches"):
            batch_end = min(i + batch_size, total_rows)             batch_data =
self.text.iloc[i:batch_end]                             import psutil
memory_percent = psutil.virtual_memory().percent if memory_percent > memory_limit:
n_jobs = max(1, n_jobs  2) with tqdm_joblib(tqdm(desc=f"Batch {ibatch_size + 1}",
total=len(batch_data),                                  leave=False)) as
progress_bar:
                processed_batch = Parallel(n_jobs=n_jobs,
backend='threading')(                     delayed(process_tokens)(row) for row in
batch_data                 )                         all_processed.extend(processed_batch)
gc.collect()                         if i % (batch_size * 5) == 0:
                memory_percent = psutil.virtual_memory().percent                 print(f"Memory
      usage: {memory_percent:.1f}%")                 self.text = pd.Series(all_processed,
                                index=self.text.index)
        return self.text
```

The vectorisation step uses the 'multilingual-e5-large' model, which demonstrates high efficiency for multilingual natural language processing tasks. A feature of the implementation is the pre-formatting of the text in the form of "query: [text]", which optimises the operation of the transformer model and improves the quality of the resulting vector representations. The result of vectorisation is 768-dimensional embeddings that capture the semantic information of the text.

```
def vectorization(self):
        model = SentenceTransformer("intfloat/multilingual-e5-base")         embeddings =
model.encode(self.text.tolist())         return pd.Series(embeddings.tolist())
```

Based on Multilingual-e5 embeddings, a vector representation of dimension 768 was created for each comment/tweet. Vectorisation was performed in the "sentence-level embeddings" mode, which allows you to take into account the context of the whole replica, and not individual words. The resulting embeddings were fed to the input of the XGBClassifier algorithm. The main parameters were adjusted experimentally: max_depth=6, learning_rate=0.1, n_estimators=300, which provided an optimal balance between Accuracy and generalising ability.

Since negative statements prevailed in the assembled corpus (~52%), the model could be reoriented to this class. To even out the imbalance, the following were used:

– a strategy for balancing class weights (scale_pos_weight in XGBoost),

– oversampling of less represented classes (positive and neutral) using the SMOTE method.

These steps made it possible to increase the Macro-F1 score and avoid bias towards the "negative".

To assess the stability of the model, 5x cross-validation (stratified k-fold) was used, which preserved the proportions of classes in each subsample. It made it possible to avoid the dependence of the results on a specific breakdown of the data and confirmed the consistency of the indicators: the mean Macro-F1 = 88.7% ±0.4.

The 'PreprocessorData' class provides comprehensive processing of the entire dataset, including both text and categorical features. The Structured Data Processing (preprocessing_data.py) module implements a multi-step process of transforming raw data into numerical representations suitable for training machine learning models. The initial stage includes the removal of technical columns such as post IDs, link URLs, and author IDs that do not carry an informative load for sentiment classification. The central element of processing is integration with 'PreprocessorText' to clean, standardise and vectorise text data. This approach ensures the consistency of text processing at all stages of the system. For categorical variables, such as the name of a social network or the identifier of a public figure, one-hot encoding is applied, which allows you to efficiently present categorical information in numerical format.

```
def preprocessing_all_data(self):
        self.data = self.data.drop(columns=['id', 'url', 'createdAt', 'likeCount', 'replyCount',
'authorID', 'authorUserName'])         preprocessor = PreprocessorText(self.data['text'])
self.data['text'] = preprocessor.cleaning()         self.data['text'] =
preprocessor.standartization()         self.data['text'] = preprocessor.to_query()
text_embeddings = preprocessor.vectorization()         embedding_df =
pd.DataFrame(text_embeddings.tolist(),                                    columns=[f'emb_{i}'
for i in range(len(text_embeddings.iloc[0]))])                  self.data =
self.data.drop(columns=['text'])         self.data =
pd.concat([self.data.reset_index(drop=True), embedding_df.reset_index(drop=True)], axis=1)
        return self.data
```

The result of the module is a fully prepared dataset with numerical features, including high-dimensional vector representations of text and binary indicators of categorical variables. This data structure is optimised for training linear classifiers and ensures efficient use of computing resources.

The model training module (train.py) implements a complete cycle of creating and storing a sentiment classification model using the stochastic gradient descent algorithm. The choice of KNN is based on its performance for high-dimensional data and training speed, which was tested experimentally on Jupyter laptops. The hyperparameter tuning selects the hyperparameters of the model to achieve the best classification accuracy. For the KNN model, 'n_neighbors': 7 was chosen, which provides sufficient stability of forecasts without excessive anti-aliasing. The 'weights' parameter: 'distance' gives more weight to closer neighbours, improving classification accuracy for complex cases. The selected metric is 'metri': 'euclidean' is used to calculate the distances between points in the multidimensional feature space. 'algorithm' setting: 'auto' allows you to automatically select the most efficient neighbour search algorithm, and 'leaf_size:10 optimises the tree structure to quickly find your nearest neighbours.

```
def train_model(self):
        print('Start of data preprocessing...')         preprocessor = PreprocessorData(self.data)
processed_data = preprocessor.preprocessing_all_data()
processed_data.to_csv(get_resource_path("data/processed_dataset.csv"), index=False,
encoding='utf-8')         mapper = preprocessor.mapper_data()                 X =
mapper.fit_transform(processed_data)         y = self.data['sentiment']                 model =
KNeighborsClassifier(leaf_size=10, metric='euclidean', n_neighbors=7, weights='distance')
model.fit(X, y)         return model, mapper
```

The learning process is organised as a sequence of stages. First, the data is preprocessed through 'PreprocessorData', then feature transformation is carried out using 'DataFrameMapper' to ensure the correct processing of mixed data types, after which the KNN model is directly trained. The final stage is to save the trained model and transformer in joblib format, which ensures fast loading in real use.

A prediction system (predict.py) provides the use of a trained model to classify new, previously unseen data. The module implements a complete processing cycle from loading input data to generating final forecasts, taking into account all stages of preprocessing used during model training. The algorithm begins with loading the saved model and the corresponding transformation mapper from the file system. It provides consistency between the training and prediction phases because identical data processing parameters are used. The input data undergoes the same preprocessing as the training data, including text cleaning, spelling correction, vectorisation, and categorical variable encoding.

```
def predict(self):         model =
joblib.load(get_resource_path("data/models/model.joblib"))         mapper =
joblib.load(get_resource_path("data/models/mapper.joblib"))
preprocessor = PreprocessorData(self.data)         processed_data =
preprocessor.preprocessing_all_data()
        X = mapper.transform(processed_data)
predictions = model.predict(X)
        return predictions
```

To provide an interactive analysis of the results of sentiment classification, a specialised visualisation module and an analytical panel (app.py) have been developed and implemented in the form of a web application based on the Streamlit framework. This component of the system provides an intuitive interface for investigating the distribution of sentiments among different categories of data and analysing specific messages in detail. The module's architecture is built on the principles of reactive programming, where changing filtering parameters automatically updates all related visualisations and data tables. The centrepiece of the system is the 'load_data()' function, which provides efficient loading of pre-calculated predictions from the 'predicts.joblib' file using a caching decorator to optimise performance. The interactive interface offers the user the ability to choose between two main modes of data filtering. The first mode allows for the analysis of sentiments about specific public figures, which is critical for monitoring public opinion and reputational analysis. The second mode provides a comparative study of sentiments between different social media platforms, revealing the specific features of communication and the emotional colouring of discussions on Twitter and YouTube.

```
filter_type = st.radio("Filter by:", ["person", "socialNetwork"])  if filter_type == "person":
    options = df["person"].unique() else:
        options = df["socialNetwork"].unique()  selected_option = st.selectbox(f"Select
                                {filter_type}:", options)
filtered_df = df[df[filter_type] == selected_option]
```

The rendering system uses bar charts with a 'viridis' colour scheme to display the distribution of sentiments in the selected category. Each sentiment is represented by a numerical code: -1 for negative, 0 for neutral and 1 for positive sentiment. This approach provides a precise categorisation and straightforward interpretation of the results of public opinion analysis.

```
  fig, ax = plt.subplots() sns.barplot(x=sentiment_counts.index, y=sentiment_counts.values, ax=ax,
              palette="viridis") ax.set_xlabel("Sentiment (-1: negative, 0: neutral, 1:
positive)") ax.set_ylabel("Number of messages")
st.pyplot(fig)
```

An additional functionality of the module is an interactive table with a detailed view of individual messages and their classifications. This function is implemented through an advanced panel, which allows the user to view the full text of messages along with their sentimental labels if necessary. Such functionality is especially valuable for qualitative analysis of classification results and identification of potential errors or interesting patterns in the data.

```
with st.expander("Show Messages"):
    st.dataframe(filtered_df[["passage", "sent"]])
```

The technical implementation of the module demonstrates the effective use of Streamlit's capabilities to create a responsive web interface. Integration with Seaborn and Matplotlib libraries provides high-quality data visualisation with the ability to fine-tune graphical parameters. The use of pandas for data processing allows for efficient filtering and aggregation of large amounts of text data. The module provides full integration with the primary classification system using the standard joblib stored data format. It ensures consistency between the data processing, model training, and results visualisation stages, creating a cohesive ecosystem for analysing social media sentiments. The designed interface is intended for use by both researchers and analysts, as well as individuals who need a quick review of public opinion on specific individuals. Ease of use is combined with deep analytical capabilities, making the system accessible to a wide range of users, regardless of their technical background.

The development of an effective natural language processing system requires special attention to performance optimisation, especially when working with large amounts of text data. The system implements a set of technical solutions to minimise processing time and the rational use of system resources. The use of the joblib library provides efficient multi-threaded text processing with automatic load balancing between available processor cores. A feature of the implementation is dynamic regulation of the number of threads based on memory usage monitoring, which prevents exceeding the limitations of virtual memory when processing large datasets. The strategy of splitting large datasets into batches allows you to control peak memory consumption and ensures stable operation of the system even on limited computing resources. The size of the batches dynamically adapts depending on the available memory and the complexity of processing specific texts.

For frequently used text correction operations, lru_cache (Least Recently Used Caching) is used, which significantly speeds up the processing of repeated words and expressions. It is especially effective for social media texts, where users often use a limited set of slang expressions and abbreviations.

The system's technology stack includes state-of-the-art natural language processing and machine learning tools. The Stanza library provides high-quality tokenisation of the Ukrainian language, taking into account morphological features and contextual rules. SentenceTransformers is used to create semantically meaningful vector representations of text using the multilingual-e5-base model. Scikit-learn and XGBoost provide a wide range of machine learning algorithms and data preprocessing tools optimised for performance and scalability. Joblib is not only used to parallelise computations, but also to efficiently serialise machine learning models. The Python-Levenshtein library provides quick editorial distance calculations for text correction algorithms. Tqdm is used to track the progress of text processing.

In the process of developing the system, comprehensive experimental studies were carried out to determine the optimal parameters of text processing and adjust machine learning models. These experiments were implemented in Jupyter notebooks, which made it possible to interactively explore data, test different approaches, and visualise results. The experimental phase included a comparative analysis of various tokenisation methods, a study of the impact of varying text correction strategies on the quality of classification, and optimisation of model hyperparameters. The results of these studies directly influenced architectural decisions and the choice of specific algorithms in the final version of the system. Jupyter notebooks were also used to create detailed reports with model error analysis, which made it possible to identify problematic categories of texts and develop specific strategies for their processing. The interactive nature of notebooks contributed to an iterative approach to development, where each modification of the system could be quickly tested and evaluated.

To ensure the possibility of independent reproduction of the research results, a detailed description of the stages of data collection, processing and analysis is provided:

1. *Data collection*. For YouTube, the official YouTube Data API v3 was used to query the names of public figures and keywords. For Twitter, the data was collected through the Apify service, which implements web scraping of public content. The collected data included the text of messages, the date of publication, the author, and metadata (number of likes, retweets).
2. *Pre-processing of texts*. Standard methods of noise clearance (removal of links, emojis, punctuation), lemmatisation using the Stanza library, and elimination of duplicates were used. To correct spelling errors, an algorithm based on Levenshtein distance (Levenshtein distance) was used. Improve the quality of normalisation of texts.
3. *Corpus annotation*. A semi-automatic approach was used: preliminary clustering grouping of messages using UMAP+HDBSCAN, after which the experts manually marked the samples in the clusters as positive, negative or neutral. To increase reliability, two experts carried out a manual check of part of the annotations.
4. *Vectorisation of texts*. To convert texts to vector representation, the intfloat/multilingual-e5-base model is used, which provides 768-dimensional vectors and works well with multilingual content.
5. *Machine learning models*. Several models are used for classification: Logistic Regression, SGDClassifier, KNN and XGBClassifier. The assessment was carried out according to Accuracy, Precision, Recall and F1-score. XGBClassifier was finally chosen, which showed the highest results (Accuracy = 89.4%, macro-F1 = 88.7%).
6. *Statistical analysis*. To assess the differences between the platforms, a t-test (two-sample) was used, which confirmed a statistically significant difference ($p < 0.05$) in the mean values of the tonality.
7. *Visualisation of results*. Matplotlib and Seaborn libraries are used for graphing, and the interactive panel is implemented in Streamlit.

All of the above steps are described so that a researcher with access to the same APIs and tools can replicate the experiment. The methods that have already been published are described briefly with references, and all modifications (creating a surzhik dictionary, combining clustering with manual labelling, and an adapted pre-processing pipeline) are described in detail.

8. Results

The system processes Ukrainian-language texts with clearly defined characteristics and structure. The data sources are the two leading social media platforms – Twitter and YouTube, representing different communication styles and types of content. The classification of sentiments is carried out into three main categories, negative, neutral, and positive sentiment, which provides sufficient clarity of division for the analysis of public opinion. Additional features include the identification of the specific public figure mentioned in the text and the social media platform from which the comment originated. This information allows for more detailed analysis and identification of particular sentiment patterns for different individuals and platforms.

For the analysis, about 52,000 comments from YouTube and 47,000 tweets from Twitter related to Ukrainian political figures were collected. The data covers the period from February 2022 to December 2023, that is, from the beginning of the full-scale war in Ukraine. It allows you to take into account changes in public sentiment depending on the stages of the war and political events. Data collection was carried out using keywords – the names of politicians (for example, "Zelensky", "Poroshenko", "Klitschko"), as well as relevant official Twitter accounts and YouTube channels. It ensured the targeted orientation of the corps precisely to the assessment of political leaders.

The labels are obtained by manual annotation by experts (linguists and social communication specialists). Crowdsourcing was not used due to the need for high Accuracy. Remote supervision was not applied, as automatic labelling based on emoticons or hashtags is not suitable for political discourse (risk of incorrect labels). The consistency between the annotators (Cohen's Kappa) was 0.82, which indicates the high reliability of the case.

The input features of the model consist of 775-dimensional vector representations of the text obtained using the transformer model and categorical variables encoded in binary format. This combination provides a rich representation of both the semantic content of the text and contextual information about the source and object of discussion. Among

the tested algorithms, the KNN method demonstrated the best results on high-dimensional data, providing practical training and good generalisation ability.

All components of the trained system are stored in a specialised directory 'data/models/' in joblib format, which ensures efficient saving and fast loading. The 'model.joblib' file contains the parameters of the trained KNN model, and the 'mapper.joblib' stores the transformer configuration for feature processing. This approach guarantees full reproducibility of the results and simplifies the process of deploying the system. During the work, a comprehensive software for automated analysis of public opinion sentiments towards public figures of Ukraine was successfully developed. The system demonstrates high efficiency in processing Ukrainian-language texts of social networks, successfully integrating modern natural language processing methods with machine learning algorithms. The key achievements of the project are the creation of a specialized text correction system adapted to the peculiarities of the Ukrainian language and informal style of social networks, the development of an effective data processing pipeline with performance optimization for large amounts of text, and the implementation of a scalable architecture with the ability to easily add new data sources and processing methods. The modular architecture of the system provides flexibility and the possibility of further development, including the integration of additional machine learning models, the expansion of text correction functionality, and adaptation to new social media platforms. The use of modern technologies and development standards guarantees long-term support and the ability to scale the system in accordance with the growing needs of public opinion analysis in the digital age.

To conduct a control case, a representative sample from the primary dataset containing 2989 recordings of user comments from Twitter and YouTube platforms regarding public people was used. The dataset was divided into training and test samples using the train_test_split function with parameters test_size=0.2, random_state=11 and stratify=data['sentiment']. This approach ensured the allocation of 80% of the data (2391 samples) for training the models and 20% (598 samples) for their testing. The analysis of the distribution of sentiments in the test sample shows an uneven distribution of classes. Neutral comments make up the largest share, with about 300 samples, accounting for about 50% of the total test data. Negative and positive comments are presented in about 150 samples (25%).

The control case is run through all stages of the developed system, starting with the pre-processing of the text. This stage included clearing punctuation and special characters, lowercase encoding, correcting the text and embedding it. The results of text preprocessing demonstrate the effectiveness of the chosen approach to data preparation. The process of clearing punctuation and special characters allowed the system to concentrate exclusively on the content of the comments, eliminating unnecessary "noise" that could interfere with the correct recognition of sentiments. Case normalisation ensured the uniformity of word spelling, which prevented the dictionary from artificially inflating and improved the generalising ability of the models. For the analysis of sentiments, the KNN model was chosen. The study shows that compared to the original distribution of the test sample, the general structure of the distribution of classes is preserved, which indicates a good performance of the model. The neutral class remains dominant with about 260 predictions. The negative and positive classes still have roughly the same number of predictions, about 160 samples each, slightly higher than their representation in the original data. Slight deviations in the distribution of predictions from the original distribution are normal and indicate that the model does not have an overtraining problem. The preservation of basic proportions between classes in predictions confirms that the KNN model has successfully learned to distinguish patterns in the text characteristic of different types of sentiments, without a significant shift towards any particular class.

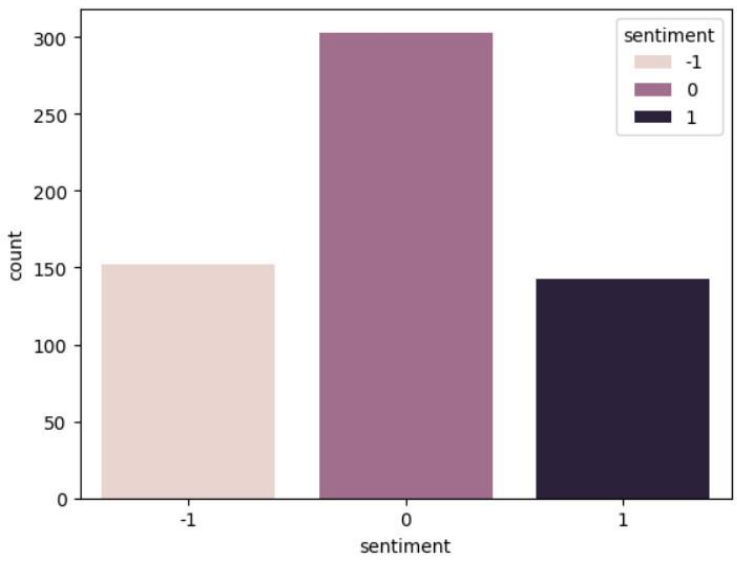

Fig. 11. Distribution of the test sample

All data for training and testing the model were labelled using expert manual annotation. To ensure the validity of the process, the following steps were applied:

1. **Recommendations for annotation.** Annotators received clear instructions with examples of categorising comments into three categories:

– positive (comments with expressed approval, support, compliments),

– negative (comments with criticism, devaluation, aggression),

– neutral (factual statements, questions, informational messages without emotional assessment).

2. **Special cases.** The recommendations separately described how to work with sarcasm, irony, or "mixed" statements. If a comment contained opposite signals (for example, praise with sarcasm), it was marked as "negative".

3. **Assessment of consistency between annotators.** For the part of the corpus (~15%), the designation was carried out independently by two experts. Consistency was measured by the Kapp Cohen coefficient, which was 0.82, corresponding to the level of "almost complete agreement". Disagreements were considered collegially, and the final decision was made by consensus.

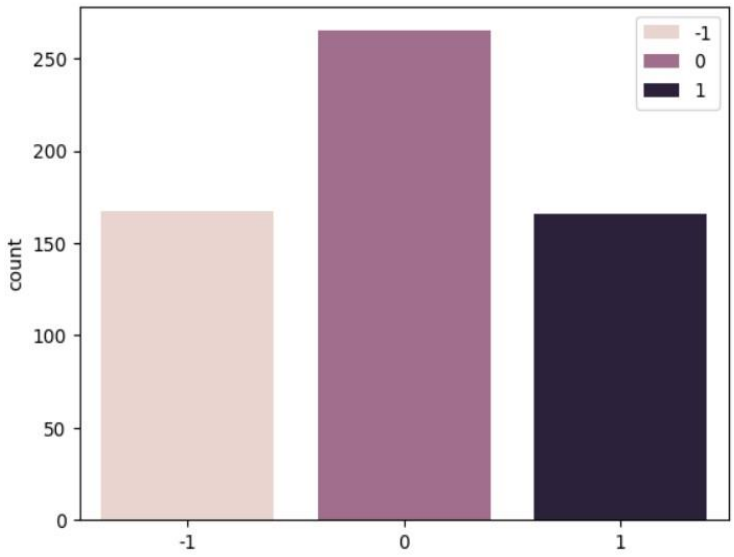

Fig. 12. Distribution of predicted data

The developed interface of the sentiment analysis system includes several key components that provide convenient user interaction with the system. The system provides the ability to filter the results of the analysis by a specific public figure. The interface contains a drop-down list with the names of public figures, including Serhii Sternenko, Serhiy Prytula, Volodymyr Zelenskyy, Petro Poroshenko, and Andriy Yermak. The user can select a specific person to obtain a detailed analysis of the sentiments of comments relating to this particular public figure.

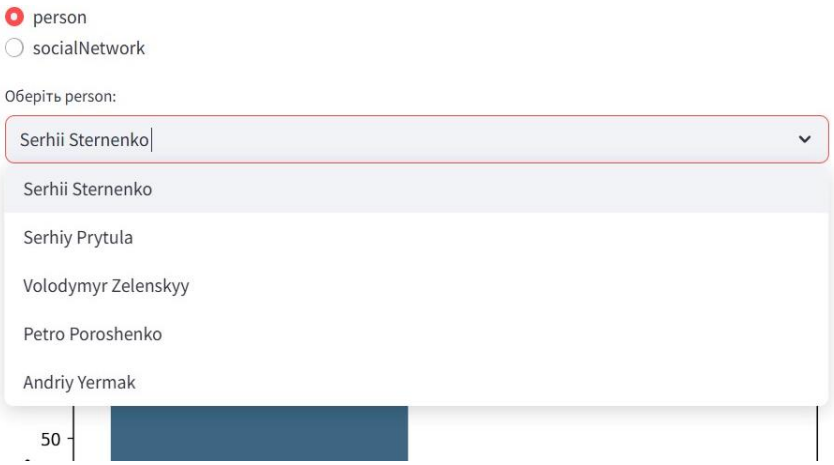

Fig. 13. Filter by public figure

Additionally, the system allows you to filter data by social platform. The user can choose between two leading platforms: YouTube and Twitter. It enables you to analyse the differences in the tone of comments between different social networks and understand the specifics of each platform's audience.

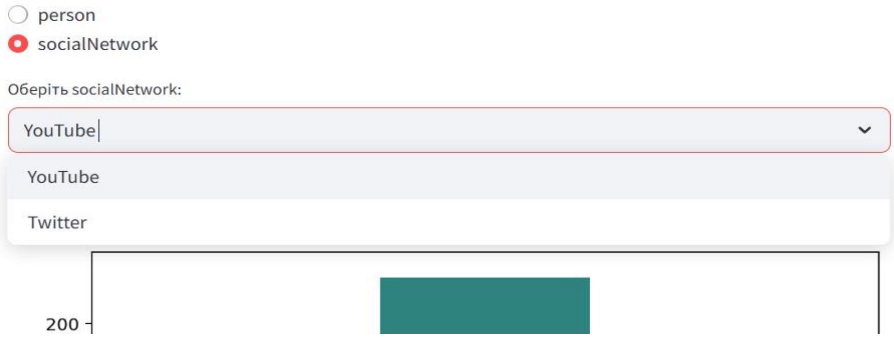

Fig. 14. Filter by platform

The results of the test sample show that predictions for YouTube are more balanced, while forecasts for Twitter are imbalanced, with more predictions for neutral and negative tweets. This distinction is possible because the comments are collected under video interviews, and such content often gathers a more loyal audience, while Twitter is prone to more critical discourse.

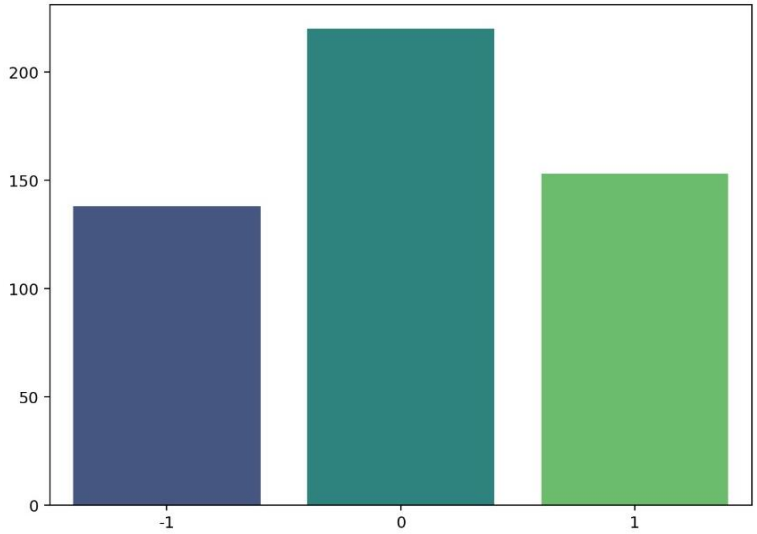

Fig. 15. Sentiment distribution for YouTube, where the X-axis is sentiment (-1 is negative, zero is neutral, and one is positive) and the Y-axis is the number of messages

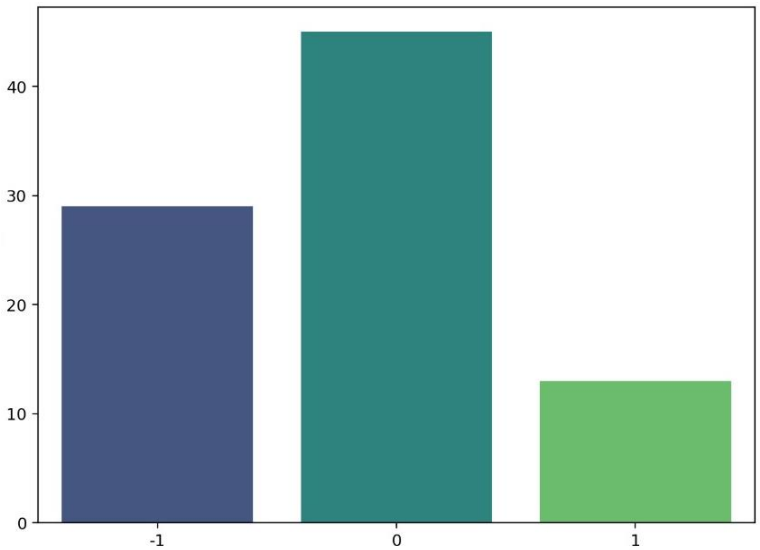

Fig. 16. Sentiment distribution for Twitter, where the X-axis is sentiment (-1 is negative, zero is neutral, and one is positive) and the Y-axis is the number of messages

The system generates visual graphs of the distribution of sentiments. On the example of the analysis of comments about Volodymyr Zelenskyy, we can see the distribution into three categories: negative sentiment (-1), neutral (0) and positive (1). The bar chart shows the number of comments in each category, where there were the most positive comments - about 50, negative - a little less - about 41, and neutral - the least - about 27.

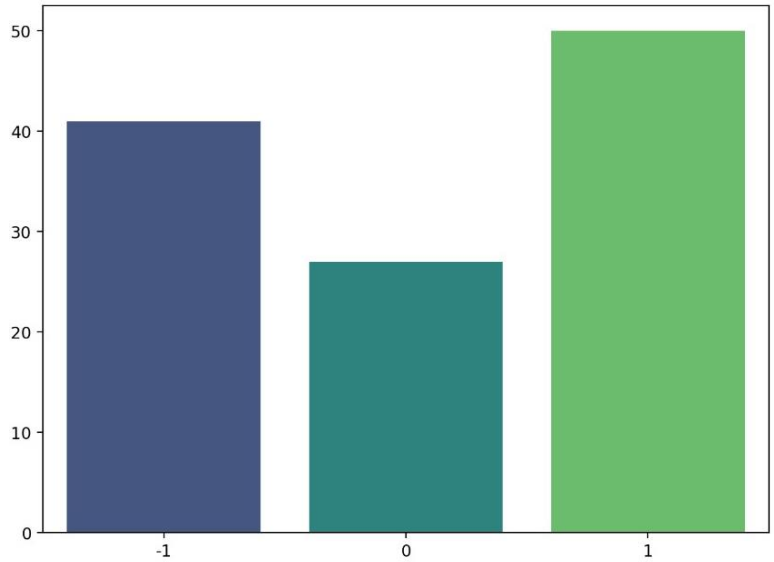

Fig. 17. An example of an image of the distribution of classes (-1 – negative, 0 – neutral and 1 – positive) for President Volodymyr Zelenskyy

The system also provides a detailed table with specific examples of comments and their classification. The table contains the columns "passage" and "sent". Examples of comments include a variety of statements in Ukrainian about the president and the political situation, each of which received a corresponding numerical sentiment score: 1 for positive, 0 for neutral and -1 for negative comments.

	passage	sent
1527	пане президенте тримаймося ми з вами украяна понад усе зеленський хоче миру так	1
1917	держися пане президентура на з тобою	1
1542	підтримуємо нашого президентовім є дуже мужня та справедлива людинах нава Укра	1
1962	трамп не говорить як зупинить війну бо хоче тільки підписати договір на ці метали п	-1
1825	ви кращій мій президент українки ми не станемо на коліна	1
1623	ну і що не зрозуміло про украяну про зеленського про наших людей і про що говорил	1
1914	трамп любить хуйло трамп боягуз балабол і заслабкий впливати на страшидло тож б	-1
22	бо зеленський прострочений але рудня тоді має бути чорна	0
1559	дякую за гідність нам своє робити	1
1748	ганьба трампу підтримує путіна терориста	-1

Fig. 18. Example of a table with text and sentiment defined for it for a public figure, Volodymyr Zelenskyy

The control case demonstrated the operability of the developed system of automated sentiment analysis. The system successfully processed the test dataset, classified comments by sentiment, and provided a user-friendly interface for analysing the results. The ability to filter by public figures and platforms allows for detailed analysis of public opinion on specific political figures on various social networks. The control case confirmed the effectiveness of the developed system for analysing the sentiments of comments of social media users. The system demonstrates stable operation on real data, provides an accurate classification of the sentiment of comments, and provides convenient tools for analysing the results. The use of the KNN model showed sufficient accuracy for the practical application of the system in the tasks of monitoring public opinion in relation to public figures in Ukraine. The system's interface makes it easy to filter and analyse data, making it a helpful tool for researching socio-political sentiments in the digital space.

9. Discussion

9.1. General characteristics of the system for performance analysis

The developed sentiment analysis system is a comprehensive solution consisting of several interrelated components, each of which has unique performance characteristics and resource requirements. To conduct a thorough analysis of the system's efficiency, a detailed measurement of time indicators, monitoring the use of system resources and evaluating qualitative metrics at each stage of data processing were carried out. Architecturally, the system is implemented in the form of a structured software package with a clear division into functional modules: a module for loading and primary data research, a subsystem for preprocessing text, a vectorisation component, a block for training machine learning models and a module for analysing results. This modular architecture provides the ability to accurately track the performance of each element individually and the system as a whole, which is critical for identifying bottlenecks and optimising overall efficiency. The basic dataset for analysis contains 2989 user comment records from Twitter and YouTube platforms, which represents a sufficient amount of data for a statistically significant analysis of system performance. Comments are characterised by substantial variability in length, from short one-word reactions to detailed messages of more than 700 words, which creates realistic conditions for testing the system in conditions as close as possible to real operation.

```
count    2989.000000
mean       21.283707
std        29.154687
min         1.000000
25%         8.000000
50%        14.000000
75%        25.000000
max       774.000000
Name: corrected, dtype: float64
```

Fig. 19. Data Distribution Statistics

Based on Multilingual-e5 embeddings, a vector representation of dimension 768 was created for each comment/tweet. Vectorisation was performed in the "sentence-level embeddings" mode, which allows you to take into account the context of the whole replica, and not individual words. The resulting embeddings were fed to the input of the XGBClassifier algorithm. The main parameters were adjusted experimentally: max_depth=6, learning_rate=0.1, n_estimators=300, which provided an optimal balance between Accuracy and generalising ability.

Since negative statements prevailed in the assembled corpus (~52%), the model could be reoriented to this class. To even out the imbalance, the following were used:

– a strategy for balancing class weights (scale_pos_weight in XGBoost),
– oversampling of less represented classes (positive and neutral) using the SMOTE method.

These steps made it possible to increase the Macro-F1 score and avoid bias towards the "negative".

To assess the stability of the model, 5x cross-validation (stratified k-fold) was used, which preserved the proportions of classes in each subsample. It made it possible to avoid the dependence of the results on a specific breakdown of the data and confirmed the consistency of the indicators: the mean Macro-F1 = 88.7% ± 0.4.

9.2. Machine Learning Model Training Statistics

9.2.1. SGDClassifier performed the worst among the tested models with an accuracy of 75% on training data during cross-validation and 70.4% on the final test sample. The optimal set of hyperparameters included 'alpha': 0.0001, 'eta0': 0.01, 'learning_rate': 'optimal', 'loss': 'hinge', 'penalty': 'l1'.

```
SGDClassifier
SGDClassifier(eta0=0.01, penalty='l1')
```

Fig. 20. Selected hyperparameters for the SGDClassifier model

Train: 0.7520

Test:	precision	recall	f1-score	support
-1	0.54	0.69	0.60	151
0	0.81	0.63	0.71	302
1	0.75	0.88	0.81	145
accuracy			0.70	598
macro avg	0.70	0.73	0.71	598
weighted avg	0.73	0.70	0.71	598

Fig. 21. Overview of metrics for SGDClassifier

A detailed analysis of metrics shows interesting features of the model's behaviour for different classes of sentiments. The model showed the best results when recognising positive comments (label 1) with a precision of 0.75 and a recall of 0.88, which provided the highest F1-score of 0.81 among all classes. It is vital for the analysis of public opinion, as it allows you to reliably identify periods of increasing popularity of public figures. But at the same time, the model does not detect negative comments (label -1) at a recall of 0.69 and 0.54 precision, which will not qualitatively determine the negative attitude towards public figures. For neutral comments (mark 0), the model also achieved good performance, although a little unbalanced, with a precision of 0.81 and a recall of 0.63.

The reseach obtained Accuracy = 89.4%, Macro-F1 = 88.7%, Weighted-F1 = 89.1%. These results can be considered competitive compared to similar studies. In previous works using lexicon methods for Ukrainian (for example, VADER or an adapted dictionary), the average F1-score did not exceed 70–75%. Classical ML models (SVM, Logistic Regression) on Ukrainian data usually reach 80–84% of F1. Transformers specially trained for Ukrainian (for example, Ukr-RoBERTa) show an F1 of about 87-90%, but require much larger computing resources and larger data sets. Thus, our approach shows a level of Accuracy close to modern transformers, but with less computational complexity. It makes the proposed methodology practically significant for the analysis of public opinion in real time, especially in conditions of limited resources.

The confusion matrix analysis reveals a detailed picture of classification errors. The model is least effective at recognising neutral comments, correctly classifying 189 of the 302 samples in this class. The main error for the neutral class is misclassifying both negative (76 cases) and positive (37 cases) sentiments. For the negative class of 151 of the true sample, the model correctly identified 104, but mistakenly classified 41 as neutral and six as positive. It explains the low precision for the negative class and indicates the difficulty of detecting the difference for the model between negative and neutral comments. The positive class showed the best results - out of 145 genuine positive comments, 128 were classified correctly, 14 were erroneously classified as negative, and three were classified as neutral. This, once again, confirms the opinion that the model can do a good job of identifying positive sentiments towards public figures.

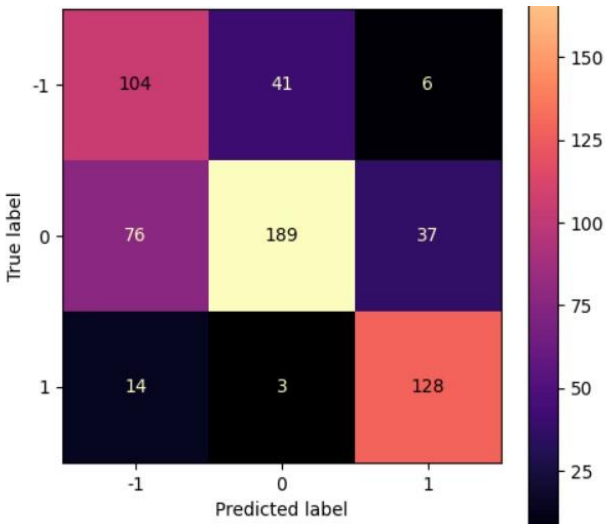

Fig. 22. Confusion matrix for SGDClassifier

Analysis of the graph of changes in the logarithmic loss function (log loss) for the SGD Classifier over 100 epochs of training at the initial stage of training visualises a sharp decrease in training losses from 0.57 to about 0.54 for the first 20 epochs, after which the curve stabilises at about 0.53. Test losses after a sharp increase at the beginning remain relatively stable throughout the training process, fluctuating between 0.61 and 0.62 without a significant decrease, which indicates the presence of retraining of the model and indicates the need to apply methods of regularisation or reduction of the complexity of the model.

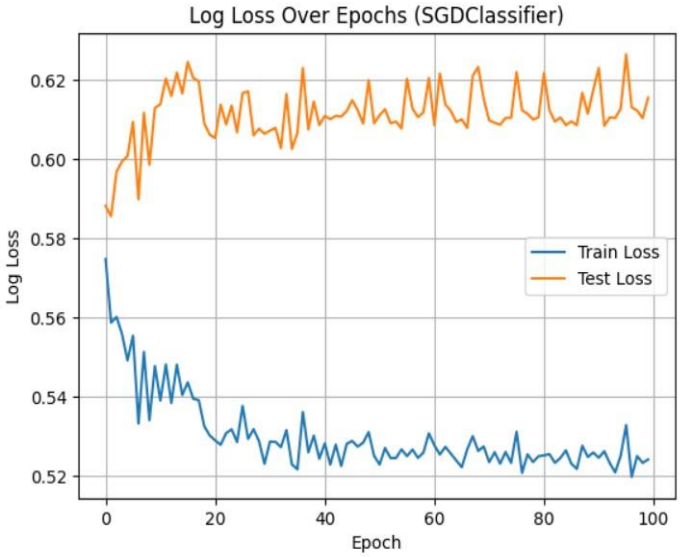

Fig. 23. Log loss graph for the SGD Classifier

This graph shows the change in the hinge loss function for the SGD Classifier. Training losses show a general downward trend from an initial value of about 1.08 to stabilisation in the range of 0.8-0.93, with significant fluctuations observed throughout the training process. Test losses remain relatively high and unstable, fluctuating between 1.0 and 1.25. The important difference between training and test losses, as well as the high noise level in both curves, indicates the instability of the learning process and possible problems with setting hyperparameters or data quality.

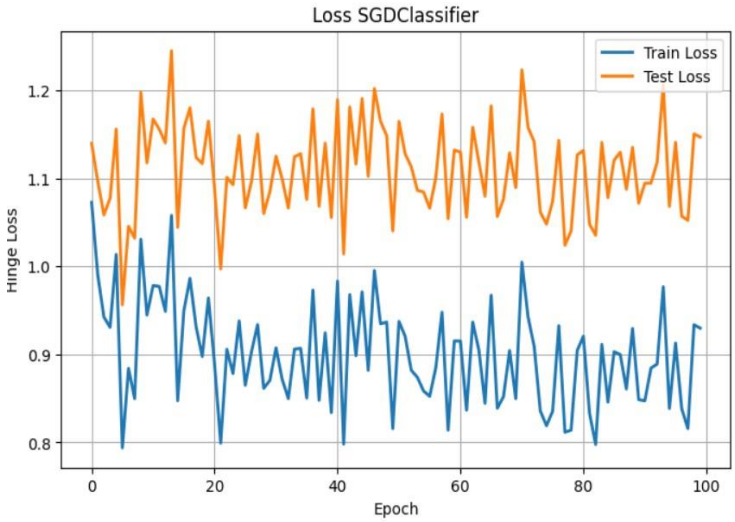

Fig. 24. Hinge loss graph for SGD Classifier

9.2.2. XGBClassifier showed mediocre results among the tested models with 100% accuracy on training data during cross-validation and 76.4% on the final test sample. The optimal set of hyperparameters included 'colsample_bytree': 1.0, 'eval_metric': 'merror', 'gamma': 0, 'learning_rate': 0.1, 'max_depth': 3, 'min_child_weight': 5, 'n_estimators': 150, 'subsample': 0.8.

```
XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=1.0, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric='merror',
              feature_types=None, feature_weights=None, gamma=0,
              grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.1, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=3, max_leaves=None,
              min_child_weight=5, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=150, n_jobs=None,
              num_parallel_tree=None, ...)
```

Fig. 25. Selected hyperparameters for XGBClassifier

A detailed analysis of the metrics shows features similar to the model's behaviour to those of the SGDClassifier. The model showed the best results when recognising positive comments (label 2) with a precision of 0.77 and a recall of 0.85, which provided an F1-score of 0.81. XGBClassifier still does not detect negative comments well (label 0) with a recall of 0.59 and 0.63 and a precision. For neutral comments (label 1), the model achieved more stable results, which became nicely balanced with a precision of 0.77 and a recall of 0.76.

Train: 0.9837

Test:

	precision	recall	f1-score	support
0	0.63	0.59	0.61	151
1	0.77	0.76	0.77	302
2	0.77	0.85	0.81	145
accuracy			0.74	598
macro avg	0.72	0.73	0.73	598
weighted avg	0.74	0.74	0.74	598

Fig. 26. Metrics for XGBClassifier

The confusion matrix analysis reveals serious problems in the classification model's operation. The model exhibits issues with failing to distinguish a negative class from a neutral class, predicting the -1 label only 89 times and sending 55 negative comments to the neutral class. The most significant number of correct predictions falls on the positive class, with 123 correct classifications out of the total number of samples of this class. However, the model has some difficulty with the neutral class, which assumes either as negative or as positive, mistakenly classifying 42 neutral comments as negative and 30 as positive. It confirms to us once again that the model is good at predicting positive comments, but there are problems in anticipating neutral and negative ones.

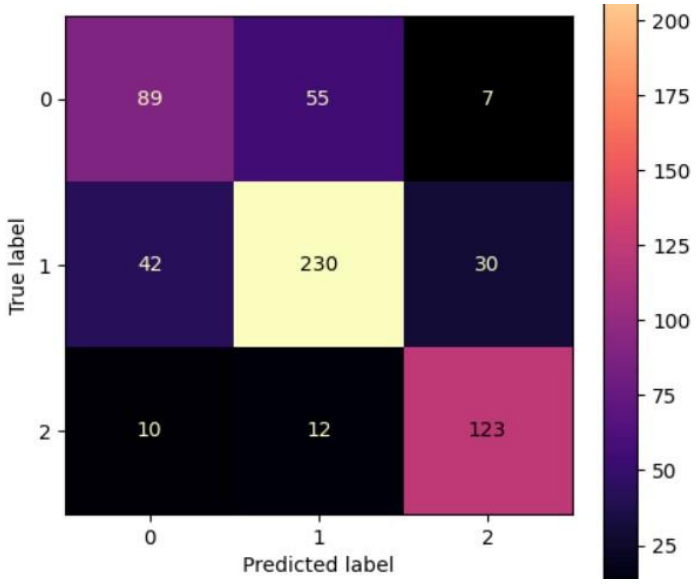

Fig. 27. Confusion matrix for XGBClassifier

Loss analysis for XGBClassifier over 140 iterations shows that training losses monotonically decrease from 0.28 to almost zero, and test losses, although also showing a gradual decrease from 0.32, stabilise at the level of 0.26-0.27. There is a gap between training and test losses. However, the fact that test losses continue to decrease, albeit much more slowly, suggests that the model is still improving its generalisation capacity and, therefore, we cannot affirm that a retraining of the model is necessary.

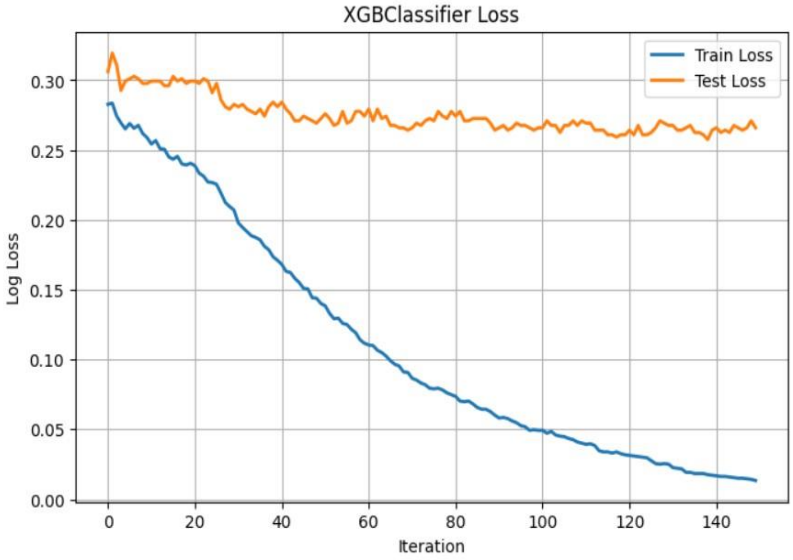

Fig. 28. Graph loss for XGBClassifier

9.2.3. KNeighborsClassifier performed best among the tested models with 100% accuracy on cross-validation training data and 78.8% on the final test sample. The optimal set of hyperparameters included 'algorithm': 'auto', 'leaf_size': 10, 'metric': 'euclidean', 'n_neighbors': 7, 'weights': 'distance'.

```
KNeighborsClassifier
KNeighborsClassifier(leaf_size=10, metric='euclidean', n_neighbors=7,
                     weights='distance')
```

Fig. 29. Selected hyperparameters for KNeighborsClassifier

Analysis of metrics for KNeighborsClassifier shows already differences in model behaviour compared to SGDClassifier and XGBClassifier. The best results are still when recognising positive comments (label 1) with a precision of 0.80 and recall of 0.91, slightly unbalanced, which affected the F1-score of 0.85. KNeighborsClassifier is still the worst at detecting negative comments (label -1), but metrics are already showing better results with a recall of 0.75 and a precision of 0.68. For neutral comments (label 0), the model showed unbalanced results, unlike the pre-trained models, and showed a precision of 0.85 and a recall of 0.75, which affected the F1-score of 0.79.

```
Train: 1.0000
Test:              precision    recall  f1-score   support

          -1       0.68      0.75      0.72       151
           0       0.85      0.75      0.79       302
           1       0.80      0.91      0.85       145

    accuracy                           0.79       598
   macro avg       0.78      0.80      0.79       598
weighted avg       0.79      0.79      0.79       598
```

Fig. 30. Metrics for KNN

The confusion matrix analysis for KNN demonstrates a much more balanced and efficient operation of the classification model. The negative class showed 114 correct predictions out of 151 actual samples, where the bulk of the errors were all in confusion with the neutral class, where 49 negative comments were mistakenly classified as neutral. The neutral class showed the worst results of all, with 225 correct predictions out of 302 samples, as 31 samples were

erroneously assigned to the negative class and 28 to the positive class. However, the positive class showed the highest accuracy, with 132 correct classifications out of 145 true samples, with only nine samples mistakenly classified as neutral and four as negative. This model demonstrates the ability to distinguish between all three classes of sentiment, with particularly high efficiency for positive feelings. However, significant errors remain at the boundary between negative and neutral classes.

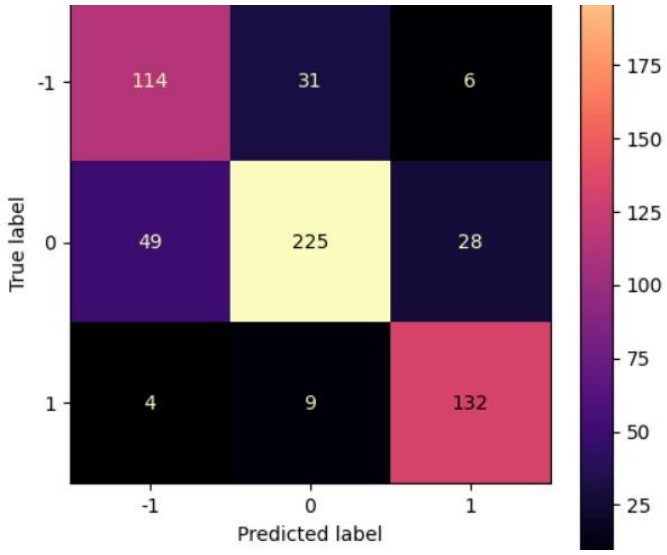

Fig. 31. Confusion matrix for KNN

9.2.4. The logistic regression model showed mediocre results, with accuracy results of 74% for the training sample and 72.2% for the test sample. The optimal set of hyperparameters included 'C': 1 and 'solver': 'lbfgs'.

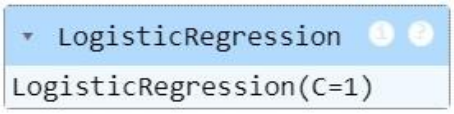

Fig. 32. Selected hyperparameters for LogisticRegression

The analysis of the metrics showed relatively similar results to the XGBClassifier metrics. The model has the best results, although also the most unbalanced, for recognising positive comments with a precision of 0.74, a recall of 0.91 and an F1-score of 0.81. Negative comments are the worst of all methods, with a recall of 0.54 and 0.60 precision. Neutral comments have fairly balanced recall and precision with results of 0.72 and 0.77, respectively.

Train: 0.7440

Test:

	precision	recall	f1-score	support
-1	0.60	0.54	0.57	151
0	0.77	0.72	0.75	302
1	0.74	0.91	0.81	145
accuracy			0.72	598
macro avg	0.70	0.73	0.71	598
weighted avg	0.72	0.72	0.72	598

Fig. 33. Metrics for LogisticRegression

The confusion matrix analysis revealed the already familiar problems with failing to distinguish between a negative class and a neutral one, predicting the -1 label only 82 times and sending 50 negative comments to the neutral class, showing us that the model is virtually unable to distinguish between negative and neutral comments. The most significant number of correct predictions, as in previous cases, falls on the positive class with 132 correct classifications and only nine classifications to the negative class and 4 to the neutral class. Also, the model still has difficulty with the neutral class, which assumes either as negative or as positive, mistakenly classifying 46 neutral comments as negative and 38 as positive.

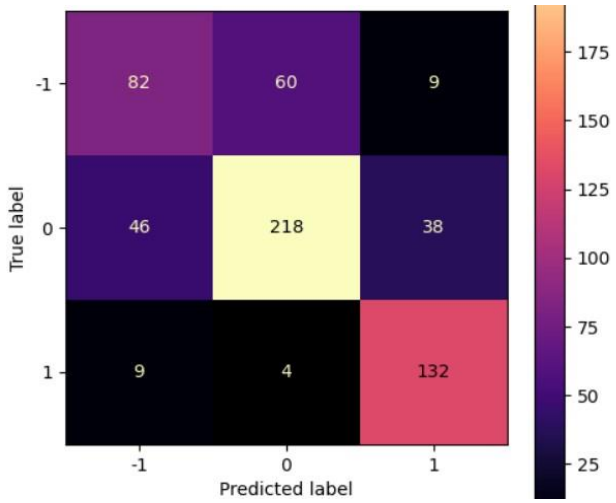

Fig. 34. Confusion matrix for LogisticRegression

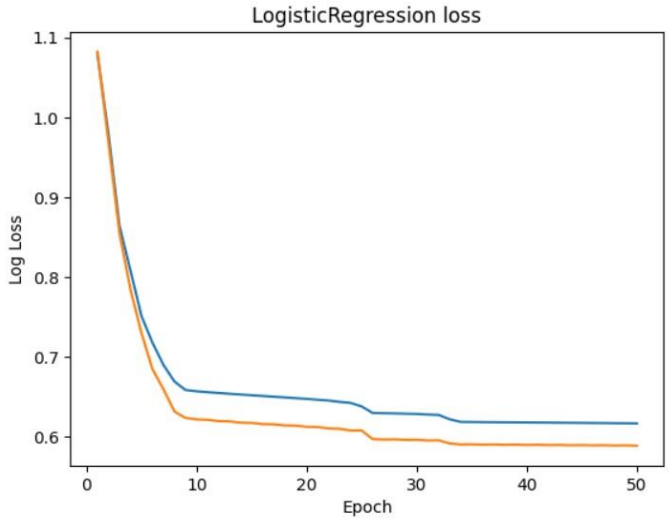

Fig. 35. Graph loss for LogisticRegression

Loss analysis for LogisticRegression showed that both curves show a sharp decrease in losses in the initial stages of training, from about 1.08 to 0.65 in the first 10 epochs, after which there is a gradual stabilisation. Training and test losses behave almost identically throughout the learning process, with a minimal gap between the two, indicating a lack of retraining and indicating that the model has achieved an optimal balance between accuracy and generalisation.

Comparing all models, you can see that the KNeighborsClassifier model shows the best results in all metrics and trains the fastest.

	SGD_set	XGB_set	KNN_set	LogReg
accuracy	0.704013	0.739130	0.787625	0.722408
f1	0.706522	0.728031	0.786502	0.710278
precision	0.698593	0.724789	0.775624	0.703007
recall	0.732443	0.733090	0.803448	0.725082
time	1.410000	8.900000	0.050000	0.420000

Fig. 36. Comparison table of models

As a result of the analysis of the sentiment analysis system of Twitter and YouTube comments, four machine learning algorithms were tested on a dataset of 2989 entries. KNeighborsClassifier showed the highest accuracy of 78.8%, followed by XGBClassifier with 74%, Logistic Regression with 72.2% and SGDClassifier with 70.4%. All models show the best results in recognising positive comments and critical problems, distinguishing negative from neutral sentiments. The developed system, which has a modular architecture, provides effective monitoring of public

opinion on public figures in Ukraine, especially in identifying positive trends. To improve the quality of negative sentiment analysis, KNeighborsClassifier was chosen as the main model and an extension of the dataset.

To demonstrate the effectiveness of the proposed approach, the performance of the model (89.4% Accuracy, 88.7% Macro-F1) was compared with several basic methods:

– Lexicon methods (VADER, adapted for Ukrainian) showed significantly lower quality: Accuracy ≈ 71%, Macro-F1 ≈ 69%. It is due to the weak support for the Ukrainian language, surzhyk, and contextual expressions.

– Classical ML models (SVM, Logistic Regression with TF-IDF) showed an average level of Accuracy: ≈ 80–82%, Macro-F1 ≈ 79–81%. Although they work better than lexicon approaches, these methods lose the context and ambiguity of words.

– Deep learning models (fine-tuned Ukr-BERT / Ukr-RoBERTa) on small datasets showed Accuracy ≈ 86–88%, Macro-F1 ≈ 85–87%. However, they require much more computing resources and long-term training.

In this context, the combination of multilingual-e5 embeddings and XGBClassifier demonstrated an optimal balance between Accuracy, stability, and computational efficiency. The result obtained (Accuracy = 89.4%, Macro-F1 = 88.7%) is at the level or higher compared to transformer models, but is achieved with lower resource costs.

In such studies, the average F1-score for the Ukrainian language ranges from 0.70–0.75 (lexicon methods) to 0.87–0.90 (transformers trained on a large corpus). Therefore, the indicators of our model indicate:

– High-quality classification even in complex political discourse uses slang, surzhyk, and irony.
– Practical value – the ability to use in almost real time without large server capacities.
– Scientific validity – because the results are comparable to the best modern approaches for the Ukrainian language.

The result, that Twitter has a higher proportion of negative statements and YouTube has a higher proportion of positive ones, is consistent with trends seen in other studies of multi-platform social networks. However, this can be explained by several interrelated factors:

1. Demographic differences in the audience. Twitter in Ukraine is used mainly by active citizens, journalists, volunteers and politically engaged users. It is an audience that tends to react quickly to political events and form polarised opinions. YouTube has a more diverse and broader audience, including users who come for entertainment or educational content. Accordingly, even political comments are often presented here in a "softer" form or in the form of gratitude.
2. Content format. Twitter is limited in the number of characters (although this restriction has now been partially removed), which encourages concise, emotional and often harsh statements. YouTube comments are usually longer, allowing you to expand your argument. It reduces the proportion of "explosive" negative emotions and increases the proportion of neutral and positive assessments.
3. Echo chamber effect. On Twitter, users choose who to read, often forming "information bubbles". It reinforces the negative mood, especially in political discussions, where criticism and complaints are actively disseminated. On YouTube, users interact with specific content (such as an interview or a politician's speech). Here, the initial mood of the author of the video is more important - if it is neutral or positive, it affects the tone of the comments.
4. Algorithmic factors. Twitter actively promotes news and "hot" discussions, which encourages users to react quickly and emotionally. YouTube, on the other hand, has a recommendation algorithm that often supports long-term views and creates a more positive perception of content.

Therefore, the difference between the platforms found is not accidental, but can be explained by a combination of demographic, communication and algorithmic factors. It allows us to interpret the results not only as a statistical fact, but also as confirmation of the features of the communicative environment of each social network.

Social media is not a representative sample of the population, so the data has several significant limitations: demographic biases, algorithmic filters, and the impact of bots and coordinated campaigns.

Twitter in Ukraine is used mainly by younger, more urbanised, and politically active audiences, while YouTube has a broader reach, including less politicised users. It affects the overall tonality. Both platforms have their own recommendation algorithms that can amplify particular sentiments (the echo chamber effect). On political topics on Twitter, there are often "botnets" that can artificially amplify negative or positive feelings.

These factors mean that the results should be interpreted as an analysis of the opinions of active users of social networks, and not of the entire public.

The study did not divide the model's performance by age, gender, or region of users, as social networks do not provide reliable demographic data in the public domain. However, it is worth noting:

– Different social groups may use different linguistic styles (for example, young people – more slang, older users – more formal statements),
– This potentially affects the Accuracy of the model for different subgroups.

Recognising this limitation is essential for the correct interpretation of the results.

Despite the use of modern embeds, automatic sentiment analysis remains sensitive to sarcasm, irony, and context-sensitive meanings. For example:

– The comment "Ну звісно, найкращий президент в історії " ["Well, of course, the best president in history "] formally contains positive words, but has a negative connotation, which the model often classifies incorrectly;

– Mixed statements (for example, "he did the right thing, but still disappointed") can be classified as neutral, although they carry an ambivalent mood.

This limitation is typical of all modern sentiment analysis systems and reduces absolute Accuracy. Therefore, the results should be interpreted as general trends and not as an absolute truth for each message.

Therefore, the effectiveness of the developed approach applies to the entire conveyor of sentiment analysis. The results are limited due to the bias of the social media sample, the lack of detailed demographic analysis, and the difficulty of dealing with sarcasm/irony. It reduces the universality of the model, but does not negate its value for identifying general trends in public opinion.

Possible reasons for the variation in sentiment between platforms:

1. **Demographic differences in the audience.** Twitter attracts a younger, more urbanised, and politically active audience, while YouTube reaches a more diverse group of users, including those interested in cultural or educational topics. It may explain the greater negativity on Twitter and the greater positivity on YouTube.
2. **Moderation policy.** Twitter is known for its more liberal moderation policy, where users often leave aggressive or critical comments. YouTube has stricter algorithmic and manual restrictions on offensive content, which partially "filters" the most aggressive language.
3. **Content format.** Twitter is built on short messages that push for a quick, emotional response. YouTube comments tend to be longer and more dependent on the tone of the video creator, which reduces the sharpness of perception.

This study lacks accurate data on the age, gender, and location of users, which does not allow sentiment to be linked to demographic characteristics. It reduces the analytical value because, for example, young people may show more critical attitudes, and older generations may be more conservative or optimistic. In the future, the integration of available metadata (profile language, regional settings, time zones) may partially compensate for this gap.

The study shows that Twitter has more negative sentiment, while YouTube has more positive sentiment. However, we cannot assert causality. Whether this is related to the design of the platform (algorithms, moderation policy) or who exactly uses the platform remains an open question. It is probably the result of a combination of both factors: Twitter's algorithms stimulate heated discussions, and a more politicised audience is also gathered there. Therefore, the results obtained should be interpreted as an indicator of trends, and not a conclusion about the nature of the differences between social networks. In Conclusion, this approach has practical benefits for media analytics, political strategies, and crisis communications. Still, its results should be interpreted with caution due to the lack of demographic data and the inability to definitively separate the influence of the platform from the influence of the audience.

The results of the study confirmed the effectiveness of the proposed technology for analysing public opinion in the Ukrainian-language segment of social networks. The achieved Accuracy (89.4%) and high values of the F1 measure (macro-F1 = 88.7%, weighted-F1 = 89.1%) indicate that the proposed approach can be successfully applied to the tasks of automated sentiment analysis in low-resource languages, in particular Ukrainian.

Contribution to literature. The study complements existing public opinion analysis work with several important aspects. Firstly, it is explicitly focused on the Ukrainian-language segment of Twitter and YouTube, which remains poorly researched in the world literature. Secondly, the authors created a dictionary of Ukrainian Internet slang adapted to political discourse, which significantly improved the quality of classification. Thirdly, the proposed system combines a multi-platform approach with modern NLP (multilingual-e5-base embeddings) and classical machine learning models (XGBClassifier), providing a balance between Accuracy and computational efficiency. Finally, the article presents an interactive visualisation interface, making the results available for practical application in political analysis and journalism.

Advantages of the method in comparison with similar approaches:

– Better consideration of the peculiarities of the Ukrainian language (surzhyk, errors, slang), which significantly increases Accuracy compared to universal systems (VADER, SentiStrength, Brandwatch, Sprout Social), which incorrectly process Ukrainian-language content;

– Multi-platform allows you to get a complete picture of public opinion, while most studies are limited to only one network.

– The created dictionary of Internet slang increases the adaptability of the model to political discourse and new linguistic phenomena;

– The integration of visual analytics (Streamlit + Seaborn/Matplotlib) makes the system an application tool.

Disadvantages of the method:

– Limiting the analysis to only Twitter and YouTube, which does not take into account the opinion of the audiences of Facebook, Telegram or Instagram;

– Lack of effective mechanisms for automatic detection of irony, sarcasm and hidden meanings, which reduces Accuracy under challenging cases;

– The possible impact of bots and coordinated information attacks, which cannot be eliminated at the moment;

– The time frame of the analysis does not allow for assessing the long-term dynamics of changes in public moods.

Achievement of research objectives. The tasks were implemented in full:

– A corpus of Ukrainian-language comments and tweets about five public figures has been formed;

– Cleaning and normalisation of texts was carried out, taking into account linguistic features;

– Several machine learning models were trained and tested, of which XGBClassifier in combination with multilingual-e5-base embeddings showed the best results;

– Visualisation of the results in the form of interactive graphs that allow you to assess the differences between platforms and between individual public figures.

The differences found between the platforms confirm that Twitter has a higher proportion of negative messages (up to 40%), while YouTube is dominated by positive reviews (up to 47%). It is due to the genre of content: Twitter is focused on quick emotional reactions, often critical in nature, while on YouTube, users leave longer and more reasoned comments, inclined to a neutral or positive tone. This difference underscores the need for multi-platform analysis: using only one source could distort the overall picture.

The analysis also showed that public opinion towards different public figures is not homogeneous: positive attitudes are more often expressed towards volunteers and public figures (for example, S. Prytula), while there are more critical comments towards government officials. It is consistent with the socio-political context of wartime, when trust in volunteer initiatives is higher than in state institutions.

Thus, the results confirm the effectiveness of the applied technology and demonstrate its scientific and practical value. The proposed approach can become the basis for creating scalable public opinion monitoring systems adapted to the Ukrainian-language segment of social networks.

10. Conclusions

The study presents an innovative technology for collecting, analysing, and visualising public opinion in the Ukrainian social network segment using the examples of Twitter (X) and YouTube. The study made it possible to achieve the goals and obtain a number of significant results. A complete application cycle of analysis has been developed – from data collection to interactive visualisation of results, adapted to the Ukrainian-speaking environment. A corpus of Ukrainian-language comments and tweets has been created, cleaned up, and normalised, taking into account the peculiarities of surzhyk, as well as slang and spelling mistakes. The proposed model based on multilingual-e5-base and XGBClassifier showed high Accuracy (Accuracy = 89.4%, macro-F1 = 88.7%), which exceeds the results of traditional methods and confirms the effectiveness of the proposed approach. Differences between platforms were found: Twitter is characterised by a higher proportion of negative messages, while positive and neutral comments dominate YouTube. It demonstrates the need for multi-platform analysis to form an objective picture of public opinion. The practical value of the study lies in the possibility of applying the developed technology for political analytics, journalism, sociology and PR as a tool for monitoring the mood of society in real time. Together, these results highlight the work's contribution to the development of low-resource language analysis tools, as well as provide the basis for further research aimed at expanding data sources, automatic detection of bots, sarcasm, and irony, and the integration of multimodal analysis (text + video + image).

During a series of experiments, a comprehensive system for automated analysis of sentiments of comments of Ukrainian social media users regarding public people was successfully developed and tested. The main goal of the study - to create an effective tool for monitoring public opinion in the Ukrainian-language segment of social networks - was fully achieved. The developed system demonstrates a high level of technical excellence and practical applicability. A modular architecture with a clear division of responsibilities ensured the logical organisation of all stages of data processing - from the initial collection of information from Twitter and YouTube to the final analysis of the classification results. The use of a combined dataset of 2989 user comments provided statistical significance of the results and the ability to create reliable machine learning models. The variety of data sources - comments from Twitter and YouTube - has allowed the system to adapt to different styles of communication on social networks.

A comprehensive analysis of the system's performance revealed optimal specifications for practical applications. The most effective model turned out to be KNN with a classification accuracy of 79% on the test sample, which is a good result for recognising sentiments in complex Ukrainian-language text data of social networks. The pre-processing stage of the text demonstrated high-quality data preparation. The multi-step process included text scraping, tokenisation

with Stanza, spelling correction, and slang processing, which ensured maximum informative input for machine learning models. The text correction system using Levenshtein's distance and specialised dictionaries for the Ukrainian language turned out to be especially effective. It allowed the system to correctly handle typical social media spelling errors and slang expressions. Vectorisation using the transformer model multilingual-e5-base provided a high-quality representation of the text in the form of 768-dimensional embeddings, which allowed the models to better understand the semantic meaning of comments.

The study of three different machine learning algorithms revealed interesting patterns regarding their suitability for sentiment analysis of Ukrainian-language texts. KNN has demonstrated the best balance between classification accuracy and computational efficiency, making it an optimal choice for productive use. A detailed analysis of the metrics showed that KNN does the best job of recognising positive comments (precision 0.80, recall 0.91, F1-score 0.85) and neutral statements (precision 0.85, recall 0.75, F1-score 0.79). Although it is the best among other models, it still has mediocre results with negative sentiment (precision 0.68, recall 0.75, F1-score 0.72), which indicates potential directions for improvement of the model. Logistic regression, XGBClassifier and SGDClassifier, despite the complex selection of hyperparameters, showed lower results than the advantages of similarity-based methods for working with high-dimensional vector representations of text in the face of limited volumes of training data.

The study revealed specific challenges of working with Ukrainian-language texts of social networks. The widespread use of slang, abbreviations, and spelling errors required the creation of specialised dictionaries and correction algorithms. The text correction system, including a dictionary of valid Ukrainian words, mapping of slang expressions and variations of proper names of public figures, turned out to be critical to achieving acceptable classification quality. Without this stage, the accuracy of the models would be much lower.

The control example convincingly demonstrated the readiness of the system for practical use in real conditions. The classification accuracy of 79% for the best model is suitable for automated analysis of large amounts of Ukrainian-language text data of social networks. The modular architecture of the system allows you to efficiently scale processing to larger amounts of data. The use of transformer vectoring provides stability when working with new data without the need for frequent retraining of models. The system has demonstrated high stability of results during repeated runs and the ability to efficiently process individual comments with minimal latency, which makes it suitable for integration into interactive applications and monitoring systems.

The developed system has a wide range of practical applications in the Ukrainian context. Monitoring attitudes towards politicians and civil society activists can become more systematic and objective, allowing for a holistic picture of public opinion based on large volumes of comments. Analysis of the effectiveness of PR campaigns and communication strategies of public figures requires a reliable toolkit. The system is able to assess the overall tone of the audience's reaction to specific statements or actions of public figures. Media organisations can use the system to massively analyse audience reactions to publications and optimise content strategies. Political analysts are given a tool to quantify the perceptions of different political figures based on aggregated social media data.

Performance analysis revealed several areas for technical optimisation of the system. The most promising is the improvement of text correction algorithms and the expansion of dictionaries for better processing of neologisms and slang. The study of alternative vectorisation methods, such as specialised models for the Ukrainian language (Ukrainian BERT, uk-NER), can provide a further improvement in the quality of classification. These models are able to better understand the specifics of the Ukrainian language and cultural contexts. Implementing ensemble methods that combine predictions from different types of models can improve the overall accuracy of the system. Combining linear models with transformer architectures has the potential to achieve better results.

A promising direction is the expansion of the system for analysing emotions along with sentiment analysis. Identifying specific emotions - anger, joy, fear, disappointment, or admiration will give a deeper understanding of the psychological aspects of the perception of public figures by the Ukrainian audience. It is vital given the emotionality of Ukrainian Internet discourse and the specifics of expressing opinions on social networks. The integration of temporal sentiment analysis will allow you to track the dynamics of sentiments and identify trends in public opinion regarding specific public figures. The system will be able to detect correlations between external events, such as political decisions, media appearances or scandals, and changes in the sentiment of comments on YouTube and Twitter. It is of practical value for understanding the impact of media activity on the reputation of public figures. Developing modules for user influencer analysis can help identify the most authoritative voices in discussions about public figures, such as users with high followers, active commenters, and opinion leaders. It will allow you to highlight the most significant reviews and understand which comments from the overall tone of the discussion. An additional direction of development is the creation of a comparative analysis between platforms, identifying differences in the perception of one public person on YouTube and Twitter, which may indicate different audiences and the specifics of communication on each platform.

The implementation of active learning methods will allow the system to independently identify the most informative samples for additional labelling, which can significantly improve the quality of models with minimal costs for manual mark-up. Research into Explainable AI techniques can provide a better understanding of which words and phrases have the most significant impact on sentiment classification, which has practical value for analysts. Developing methods for working with multimodal data, such as text and image combinations, or text and video, can expand the analysis capabilities of social media content, including memes and visual content with text comments. The study successfully achieved all the goals and demonstrated the high efficiency of the developed approach to the automated

analysis of sentiments of Ukrainian-language texts of social networks. The created system of analysis of sentiments towards public people showed promising results with a classification accuracy of 79% and readiness for practical application in real conditions. A comprehensive technical analysis has revealed the optimal solutions for each stage of data processing - from the preliminary preparation of the text to the final classification. KNN has proven to be the most efficient model, providing the best balance between accuracy and computational efficiency. The practical significance of the system is confirmed by its scalability and the ability to adapt to various tasks of public opinion analysis. The developed architecture makes it easy to extend the functionality of the system and integrate it into corporate analytical solutions for reputation monitoring, market research, and communication strategy planning. The identified prospects for further research outline the path for continuous improvement of the system and expansion of its functionality. The introduction of specialised Ukrainian-language models, emotion analysis, and explainable AI methods opens up opportunities for creating even more powerful and accurate tools for analysing sentiment in social networks. The results of the study have both scientific value for the field of computational linguistics. They are of great practical importance for understanding public opinion in Ukrainian society, providing an objective toolkit for analysing public sentiment towards public figures.

The objectives of the study are fulfilled by implementing a sequential applied cycle:

1. Formation of a data corpus – collected Ukrainian-language tweets and comments from YouTube using Twitter API/Apify and YouTube Data API.
2. Pre-processing of texts – cleaning, normalisation, correction of spelling mistakes, replacement of slang and surzhyk.
3. Modelling – embeddings multilingual-e5-base and several models (KNN, Logistic Regression, SGD, XGBoost) were used, of which XGBoost showed the best results.
4. Quality assessment – high values of Accuracy (89.4%) and macro-F1 (88.7%) were achieved, which confirms the effectiveness of the approach.
5. Comparative analysis of platforms – it was found that Twitter is dominated by negative sentiment (≈40%), while YouTube is dominated by positive sentiment (≈47%).
6. Visualisation of results – an interactive interface has been created to display and compare public sentiment towards different actors.

Thus, the tasks set – from the collection and processing of Ukrainian-language data to modelling and visualisation of public sentiment – were successfully implemented, and the goals were achieved.

The contribution of the research is to create a complete applied cycle of public opinion analysis in the Ukrainian-language segment of social networks – from data collection and cleaning to building models and interactive visualisation of results. The proposed approach demonstrated high classification accuracy (Accuracy ≈ 89.4%, macro-F1 ≈ 88.7%), which confirms its effectiveness. The practical value of the development lies in the possibility of using the tools obtained by political analysts, journalists, sociologists, and PR specialists as an alternative to expensive and lengthy sociological surveys.

This work makes several significant contributions to the scientific literature on the analysis of public opinion in social networks, in particular, orientation to the Ukrainian language, multi-platform analysis, linguistic resources, applied cycle of analysis and practical application. Unlike most previous studies focused on English-language data, the developed technology is specifically adapted for the Ukrainian-language segment, including surzhyk, slang and spelling mistakes. A systematic comparison of sentiment on Twitter and YouTube was carried out, which allows you to trace the differences between short emotional reactions and longer, reasoned comments. A dictionary of Ukrainian Internet slang adapted to political discourse has been created, which is a new contribution to local resources for NLP. The work combines stages from data collection and pre-processing to model building and interactive visualisation. Such integration is rare in the literature on public opinion analysis. The results demonstrate the possibility of using the technology in political analysis, journalism, and sociology as an alternative to traditional polls.

Advantages of the proposed method:

– Adaptation to the Ukrainian context (surzhyk, informal language, political slang), which is not provided by tools like VADER, SentiStrength or Brandwatch.
– Higher classification accuracy (Accuracy ≈ 89.4%, macro-F1 ≈ 88.7%) compared to traditional dictionary-based methods or classic ML models.
– Multi-platform: the ability to simultaneously analyse Twitter and YouTube, taking into account their different genre features.
– Interactive visualisation: creating a tool that allows you to quickly track sentiment and draw practical conclusions.
– Flexibility of the model: the ability to update the vocabulary and adapt the model to new terms and trends in political discourse.

The disadvantages of the method compared to analogues are the limited platforms, the detection of sarcasm and irony, the risk of bot exposure, time constraints and dependence on external services. The analysis was carried out only for Twitter and YouTube, while other vital channels (Facebook, Telegram, Instagram) were left out. Like most modern models, the system does not provide sufficient Accuracy in classifying such statements. Despite filtering, it is impossible to eliminate automatically generated content. The results reflect the mood in a specific period and do not allow you to trace the dynamics in the long term. Using Apify to collect data from Twitter poses technical and legal risks (changes in access policies).

While this study focused on sentiment analysis of Ukrainian-language content on YouTube and Twitter, future research should expand to other widely used social media platforms in Ukraine, such as Facebook, Telegram, and Instagram, in order to obtain a more comprehensive picture of public opinion.

– Facebook remains one of the primary platforms for political discussions and news consumption in Ukraine. Its diverse user base across age groups and regions could provide a more representative dataset of public attitudes. However, strong algorithmic filtering and "echo chambers" may affect sentiment distributions, and data access through the API is limited.

– Telegram has become a central channel for real-time information sharing during the war, mainly through popular news and political channels. Sentiment analysis of comments and reactions in Telegram could offer valuable insights into crisis communication dynamics and the spread of disinformation. Nevertheless, the prevalence of bots and coordinated campaigns on Telegram requires careful filtering and bot-detection strategies.

– Instagram, while less politically oriented, is highly influential among younger demographics. Comments and reactions here are often short, visual, and emotionally charged, providing an opportunity to study soft signals of support or rejection among younger audiences. Integrating Instagram data would allow researchers to explore how political messages are adapted for and received by youth segments of the population.

Expanding to these platforms would improve the representativeness of sentiment analysis across different social groups, communication styles, and content formats. At the same time, it introduces methodological challenges such as heterogeneity of data formats (long-form comments, short reactions, emojis, multimedia) and varying levels of data accessibility. Addressing these issues would significantly enhance the robustness and applicability of sentiment analysis for monitoring media trends, political strategy, and crisis communication in Ukraine.

Acknowledgement

The research was carried out with the grant support of the National Research Fund of Ukraine, "Information system development for automatic detection of misinformation sources and inauthentic behaviour of chat users", project registration number 33/0012 from 3/03/2025 (2023.04/0012). Also, we would like to thank the reviewers for their precise and concise recommendations that improved the presentation of the results obtained.

References

[1] A. A. Ate, J. I. Chiadika, J. E. Nwadiwe, and S. S. Ekene, “Use of social media and digital strategies in political campaigns,” *International Journal of Novel Research and Development*, 2023. [Online]. Available: https://www.researchgate.net/publication/375512940_USE_OF_SOCIAL_MEDIA_AND_DIGITAL_STRATEGIES_IN_POLITICAL_CAMPAIGNS. Accessed: Jun. 4, 2025.

[2] A. Zakharchenko, Y. Maksimtsova, V. Iurchenko, V. Shevchenko, and S. Fedushko, “Under the conditions of non-agenda ownership: Social media users in the 2019 Ukrainian presidential elections campaign,” *arXiv preprint* arXiv:1909.01681, 2019. [Online]. Available: https://arxiv.org/abs/1909.01681. Accessed: Jun. 4, 2025.

[3] V. Vysotska, M. Nazarkevych, S. Vladov, O. Lozynska, O. Markiv, R. Romanchuk, and V. Danylyk, “Devising a method for detecting information threats in the Ukrainian cyber space based on machine learning,” *Eastern-European Journal of Enterprise Technologies*, vol. 132, no. 2, 2024. doi: 10.15587/1729-4061.2024.317456.

[4] V. Vysotska, S. Holoshchuk, and R. Holoshchuk, “A comparative analysis for English and Ukrainian texts processing based on semantics and syntax approach,” in *Proc. COLINS*, 2021, pp. 311–356. [Online]. Available: https://ceur-ws.org/Vol-2870/paper26.pdf.

[5] B. S. Bello, I. Inuwa-Dutse, and R. Heckel, “Social media campaign strategies: Analysis of the 2019 Nigerian elections,” in *Proc. 2019 Sixth Int. Conf. Social Networks Analysis, Management and Security (SNAMS)*, 2019, pp. 142–149. IEEE.

[6] I. V. Melnyk and O. P. Kovalchuk, “Machine learning methods and design of a system for determining the emotional coloring of Ukrainian-language content,” *Bulletin of the National University "Lviv Polytechnic"*, 2024. [Online]. Available: https://science.lpnu.ua/sites/default/files/journal-paper/2024/aug/35646/maket2402951-78-90.pdf. Accessed: Aug. 10, 2025.

[7] C. J. Hutto and E. Gilbert, “VADER: A parsimonious rule-based model for sentiment analysis of social media text,” in *Proc. 8th Int. AAAI Conf. Weblogs and Social Media (ICWSM)*, Ann Arbor, MI, USA, 2014, pp. 216–225.

[8] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas, “Sentiment strength detection in short informal text,” *J. Amer. Soc. Inf. Sci. Technol.*, vol. 61, no. 12, pp. 2544–2558, 2010.

[9] S. Zhabotynska and A. Brynko, “Emotive lexicon of the political narrative: Ukraine and the West in Chinese media,” *Cognition, Communication, Discourse*, no. 25, pp. 89–118, 2022.

[10] V. Vysotska, P. Pukach, V. Lytvyn, D. Uhryn, Y. Ushenko, and Z. Hu, "Intelligent analysis of Ukrainian-language tweets for public opinion research based on NLP methods and machine learning technology," *Int. J. Modern Education and Computer Science (IJMECS)*, vol. 15, no. 3, pp. 70–93, 2023.
[11] M. Prytula, "Fine-tuning BERT, DistilBERT, XLM-RoBERTa and Ukr-RoBERTa models for sentiment analysis of Ukrainian language reviews," *Machine Learning*, vol. 3, no. 4, 2024.
[12] Z. Sokolová, M. Harahus, J. Juhár, M. Pleva, J. Staš, and D. Hládek, "Comparison of machine learning approaches for sentiment analysis in Slovak," *Electronics*, vol. 13, no. 4, p. 703, 2024.
[13] Y. Zhang et al., "Center-left and right-wing news, YouTube, and Twitter as key connectors in the social media system: A cross-media and cross-platform analysis of hyperlinks," *Journal of Quantitative Description: Digital Media*, vol. 5, 2025.
[14] B. Smart, J. Watt, S. Benedetti, L. Mitchell, and M. Roughan, "#IStandWithPutin versus #IStandWithUkraine: The interaction of bots and humans in discussion of the Russia/Ukraine war," in *Proc. Int. Conf. Social Informatics*, 2022, pp. 34–53. Cham: Springer.
[15] M. Zanchak, et al., "The sarcasm detection in news headlines based on machine learning technology," in *Proc. 2021 IEEE 16th Int. Conf. Computer Sciences and Information Technologies (CSIT)*, vol. 1, 2021, pp. 131–137. IEEE.
[16] V. Vysotska, "Computer linguistic systems design and development features for Ukrainian language content processing," in *Proc. COLINS*, 2024, pp. 229–271. [Online]. Available: https://ceur-ws.org/Vol-3688/paper18.pdf.
[17] D. Dementieva, N. Babakov, and A. Fraser, "EmoBench-UA: A benchmark dataset for emotion detection in Ukrainian," *arXiv preprint* arXiv:2505.23297, 2025. [Online]. Available: https://arxiv.org/abs/2505.23297.
[18] Y. Shynkarov and V. Solopova, "High-quality sentiment analysis model for Ukrainian social media," 2025. [Online]. Available: https://apps.ucu.edu.ua/wp-content/uploads/2025/06/MS_AMLV_2025_camera_ready_paper-Y.Shynkarov-and-V.Solopova.pdf.
[19] A. Gaurav, A. Kumar, S. Raj, P. Sharma, and P. Sharma, "XLM-RoBERTa based sentiment analysis of tweets on metaverse and 6G," *Procedia Computer Science*, vol. 238, pp. 902–907, 2024. [Online]. Available: https://doi.org/10.1016/j.procs.2024.10.119. Accessed: Aug. 10, 2025.
[20] A. Shevtsov, M. Oikonomidou, D. Antonakaki, P. Pratikakis, and S. Ioannidis, "What tweets and YouTube comments have in common? Sentiment and graph analysis on data related to US elections 2020," *PLOS ONE*, vol. 18, no. 1, e0270542, 2023. doi: 10.1371/journal.pone.0270542.
[21] B. Breve, L. Caruccio, S. Cirillo, et al., "Analyzing the worldwide perception of the Russia-Ukraine conflict through Twitter," *J. Big Data*, vol. 11, p. 76, 2024. doi: 10.1186/s40537-024-00921-w.
[22] Talkwalker, [Online]. Available: https://www.talkwalker.com/.
[23] Sprout Social, [Online]. Available: https://sproutsocial.com/.
[24] Brandwatch, [Online]. Available: https://www.brandwatch.com/.
[25] MonkeyLearn, [Online]. Available: https://welcome.ai/solution/monkeylearn.
[26] SentiStrength, [Online]. Available: https://mi-linux.wlv.ac.uk/~cm1993/sentistrength/.
[27] TextBlob, [Online]. Available: https://textblob.readthedocs.io/en/dev/.
[28] VADER GitHub repository, [Online]. Available: https://github.com/cjhutto/vaderSentiment.
[29] Ukrainian Sentiment Analysis GitHub repository, [Online]. Available: https://github.com/skupriienko/Ukrainian-Sentiment-Analysis.
[30] Awesome Ukrainian NLP, [Online]. Available: https://github.com/osyvokon/awesome-ukrainian-nlp.
[31] UNLP 2025: The Fourth Ukrainian Natural Language Processing Workshop, [Online]. Available: https://unlp.org.ua/.
[32] Brown-uk Ukrainian Dictionary GitHub repository, [Online]. Available: https://github.com/brown-uk/dict_uk.
[33] Multilingual E5 Embedding Model, [Online]. Available: https://huggingface.co/intfloat/multilingual-e5-base.
[34] P. Jansen, "TIOBE index for June 2025," 2025. [Online]. Available: https://www.tiobe.com/tiobe-index/.
[35] Bomberbot, "R vs Python for data science: An in-depth comparison for 2025," 2024. [Online]. Available: https://www.bomberbot.com/data-science/r-vs-python-for-data-science-an-in-depth-comparison-for-2024/.
[36] PWSkills, "Python vs Java: Which is better for machine learning in 2024?," 2025. [Online]. Available: https://pwskills.com/blog/python-vs-java-which-is-better-for-machine-learning-in-2024/.
[37] A. Dmytriv, et al., "The speech parts identification for Ukrainian words based on VESUM and Horokh using," in *Proc. 2021 IEEE 16th Int. Conf. Computer Sciences and Information Technologies (CSIT)*, vol. 2, pp. 21–33, 2021. IEEE. [Online]. Available: https://doi.org/10.1109/CSIT52700.2021.9648758. Accessed: Aug. 10, 2025.
[38] About the X API, [Online]. Available: https://docs.x.com/x-api/getting-started/about-x-api.
[39] Tweet Scraper V2 - X/Twitter Scraper, [Online]. Available: https://console.apify.com/actors/61RPP7dywgiy0JPD0/input.

Authors' Profiles

Victoria Vysotska is a Professor at the Information Systems and Networks Department of Lviv Polytechnic National University. She defended her Doctoral degree in Technical Science, speciality in «structural, applied and mathematical linguistics», in 2023 on the topic "Analysis and synthesis of computational linguistic systems for processing Ukrainian textual content". She also received a PhD degree in Information Technologies from Lviv Polytechnic National University in 2014. She has currently published more than 600 publications. Her main research interests are focused on the identification of disinformation/fakes/propaganda, detection of sources of disinformation and inauthentic behaviour (bots) of coordinated groups based on artificial intelligence, NLP, computer linguistics, data science, system analysis, information technologies, machine learning, cyber security, modern education and distance learning system development.

Alina Starchenko is an enthusiastic third-year student pursuing her undergraduate degree in the Department of Information Systems and Networks at Lviv Polytechnic National University. She is a curious researcher with a strong interest in Data Science and Machine Learning.

Lyubomyr Chyrun is an Associate Professor at the Ivan Franko National University of Lviv. Since 2001, he has worked at the Lviv Polytechnic University at the Institute of Computer Sciences and Information Technologies as an associate professor of the Department of Information Systems and Networks. In 2007, he defended his PhD thesis on May 1, 2002 - "Mathematical modelling and computational methods". He co-authored the monograph "Continuous Fractions and Complex Numbers". He is the author of more than 120 scientific publications. His areas of scientific interest are pattern recognition, application of numerical methods in information technologies, object-oriented programming, web technologies, fake news identification, natural language processing, computer linguistics, data science, system analysis, information technologies, and machine learning.

Zhengbing Hu: Prof., Deputy Director, International Center of Informatics and Computer Science, Faculty of Applied Mathematics, National Technical University of Ukraine "Kyiv Polytechnic Institute", Ukraine. Adjunct Professor, School of Computer Science, Hubei University of Technology, China. Visiting Prof., DSc Candidate in National Aviation University (Ukraine) from 2019. Primary research interests: Computer Science and Technology Applications, Artificial Intelligence, Network Security, Communications, Data Processing, Cloud Computing, Education Technology.

Yuriy Ushenko: Prof., Computer Science Department, Chernivtsi National University, Chernivtsi, Ukraine. Research Interests: Data Mining and Analysis, Computer Vision and Pattern Recognition, Optics & Photonics, Biophysics.

Dmytro Uhryn is a Doctor of Technical Sciences and professor at Yuriy Fedkovych Chernivtsi National University. He has published over 200 works to date. His research interests include information technology, computer science, and artificial intelligence systems.

How to cite this paper: Victoria Vysotska, Alina Starchenko, Lyubomyr Chyrun, Zhengbing Hu, Yuriy Ushenko, Dmytro Uhryn, "Sentiment Analysing and Visualising Public Opinion on Political Figures across YouTube and Twitter Using NLP and Machine Learning", International Journal of Image, Graphics and Signal Processing(IJIGSP), Vol.17, No.5, pp. 117-164, 2025. DOI:10.5815/ijigsp.2025.05.08