University of Bonn

CAISA lab

Introduction to Natural Language Processing

(Winter Semester 2024/2025)

Project Report of Team 23

Performance Comparison of Different ML Architectures for Narrative Classification on Ukraine-Russian War and Climate Change Topics

Group Members:

Zhastay Yeltay	50326844
Jinming Yu	3215414
Patrik Gümpel	3246896
Al-Baraa Mohammed	50002520
Elfi Rosenbaum	50246036

March 28, 2025

CAISA Lab
https://caisa-lab.github.io/

UNIVERSITÄT BONN

Contents

1. Introduction

The project aims to analyze the textual content of media articles to classify underlying narrative patterns [1]. Natural Language Processing techniques will be employed to detect and classify propaganda narratives in media articles related to climate change and the Ukraine conflict [1]. Focusing on analyzing the textual content of articles to detect phrases and patterns that align with one or more propaganda narrative categories defined in a comprehensive narrative taxonomy. Articles may include multiple labels, as narratives often overlap. Furthermore, each narrative label is subdivided into finer-grained sub-narratives, enabling a deeper understanding of the nuances within the media coverage. To achieve this, we leverage a labeled dataset of annotated media articles covering two primary domains:

- **Ukraine-Russia War (URW):** Narratives such as "Russia as the victim", "discrediting Ukraine" and "blaming the West" [1].
- **Climate Change (CC):** Narratives including "criticism of the climate movement", "downplaying climate change" and "hidden plots" [1].

The dataset will be analyzed extensively through Exploratory Data Analysis (EDA) [2], examining features such as narrative label distribution, sentence length, vocabulary diversity, and topic prevalence. Visualization tools like frequency distributions and word clouds will highlight common themes in the data. The primary objective of this research is to evaluate and compare the performance of three distinct model categories with 5 different models in classifying different narratives:

1. **Traditional Machine Learning Models:** Naive Bayes and Random Forests will be employed.
2. **Deep Learning Models:** Leveraging dense word embeddings (e.g., Word2Vec, GloVe), models such as Neural Networks (NN) and Long Short Term Memory Recurrent Network (LSTM) will be implemented.
3. **Transformer-Based Models:** Using pre-trained transformer models like BERT and Llama.

We aim to assess the effectiveness of these approaches in achieving high classification performance, focusing on the F_1-Score as the primary evaluation metric. Our goal is to determine which models or techniques are most effective for detecting propaganda narratives, not only in the context of the Ukraine-Russia war but also for other domains like climate change media coverage.

Research Questions

1. How do different machine learning and deep learning models perform in classifying propaganda narratives in media articles related to the Ukraine-Russia war and climate change?
2. What is the impact of using transformer-based models, such as BERT, on the accuracy and effectiveness of detecting nuanced propaganda narratives compared to traditional machine learning and deep learning models?

2. Related Work

Research about online messaging and propaganda has gained significant recognition over the past years because digital platforms develop rapidly. Academics have studied the underlying emotional patterns found in different forms of online communication to detect secretive motives. Research on propaganda has gained prominence since this technique enables subtle manipulation in public perception towards multiple aspects including events and social groups. This part examines important findings which describe modern understanding of these events.

A current research project demonstrates how analyzing big data can reveal patterns within social media datasets. The research by Zeng et al. (2021) examined a large dataset through which they assessed misleading post origins and propagation speed across social media networks [3]. The authors applied statistical methods to discover groups of connected messages through their quantitative methodological framework. The investigators discovered that AI systems can generate valuable information about how propaganda content stays alive and spreads on Twitter and Facebook platforms.

Li et al. (2022) proposed another research approach to identify propagandistic devices using sophisticated computational techniques in their work [4]. The authors established a technological framework which analyzes texts to identify specific rhetorical techniques from name-calling to loaded expressions. The adaptable system suits various languages and cultural settings and functions as a crucial basis for upcoming cross-national research. The paper establishes sophisticated linguistic detection methods by analyzing the growth of bot use and coordinated propaganda campaigns.

Multiple research studies show that joining computational methods with social science research methods produces more complex and comprehensive understanding than standing alone. The detection process benefits from mathematical evaluations and artificial intelligence yet experts are vital for applying context and subtle meanings to potentials. Interdisciplinary cooperation becomes vital because historical and sociopolitical circumstances determine the way propaganda influences people [5].

The current academic progress has potential untapped potential to increase capacity in handling multilanguage and cross-cultural evaluation approaches. The applicability of existing models extends only to English-language sources since they receive primary training on this specific linguistic foundation [6]. Further research needs to develop real-time methods for detection and response given online conditions that can transform over short time periods. The foundation created through these works enables further research development to perfect detection system capabilities and practical intervention frameworks against deceptive online content.

3. Data Exploration and Preprocessing

This section provides an in-depth exploration of the dataset, analyzing its composition, linguistic diversity, and distribution of narratives and subnarratives. It also details the preprocessing steps applied to clean and standardize the data, as well as the techniques used to address class imbalance, ensuring a more balanced representation of underrepresented subnarratives for improved classification performance.

Data Exploration

The dataset consists of annotated media articles with labels indicating types of propaganda narratives. Some examples include:

- **CC (Climate Change)**: Criticism of climate movement, downplaying climate change, hidden plots, etc.
- **URW (Ukraine-Russia War)**: Russia as the victim, discrediting Ukraine, blaming the West, etc.

Dataset exploration involves understanding the distribution of these labels, sentence length, vocabulary richness, and topic diversity across the dataset. We can visualize this by creating frequency distributions of label occurrences or generating word clouds to explore the most common terms [1]. For example, articles in the `URW` category may contain frequent references to "Russia", "Ukraine", and "the west" while `CC` articles might discuss terms like "climate change", "melting", "global elites" and others. To understand the data, we performed an in-depth exploratory data analysis (EDA) that is included in Table 1.

Language	**Taxonomy**	**Documents**	**Tokens**	**Vocab. Size**	**Avg. Doc Length**	**Narrative**	**Subnarratives**
English	**URW**	128	62004	14691	484	12	44
	CC	102	51030	12524	500	10	41
	Other	169	77302	16635	457	1	1
Portuguese	**URW**	208	77610	13019	373	11	41
	CC	165	68621	14691	415	9	34
	Other	27	9901	3527	266	1	1

Table 1: Statistical Measures of the documents of each main label (URW, CC, or Other) partitioned by the language of the documents.

As shown in Figure 1, the English text embeddings appear closely intermixed, which may pose challenges for classification approaches. In contrast, the Portuguese embeddings are distributed in more clearly separable clusters, suggesting that subnarratives in Portuguese texts could be detected with relative ease[1].

The dataset contains news articles in English and Portuguese about the Ukraine-Russia War (URW) and Climate Change (CC), plus other subnarratives. PCA [7, 8] and t-SNE [9] plots show that English-text embeddings mix a lot, with little cluster separation. This overlap means classification models might struggle to tell apart the subnarratives in English articles[1].

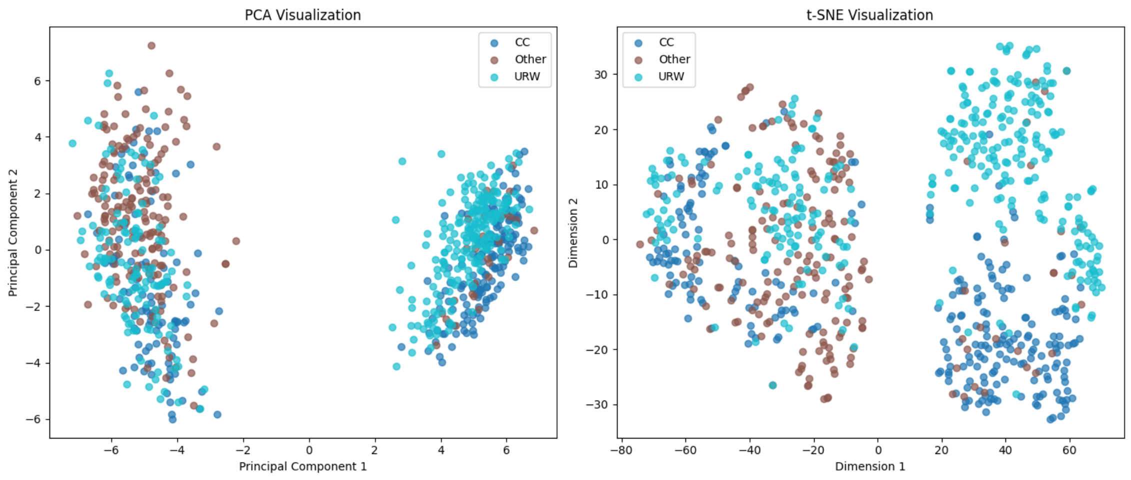

Figure 1: Two-dimensional projections of article embeddings using PCA (left) and t-SNE (right), colored by subnarrative label (URW, CC, or Other)

A complementary perspective on the dataset's subnarratives can be seen in Figure 2, which provides a graph-based representation of the relationships among various classes [10, 11]. In this visualization, certain subnarratives appear tightly interconnected, often suggesting overlapping themes or shared linguistic markers. By contrast, other subnarratives remain more peripheral, indicating fewer conceptual or lexical commonalities. This structure-based view underscores the complexity of identifying narratives that may partially overlap, while also highlighting clusters of topics that co-occur frequently. Consequently, models trained on this dataset must learn to disentangle closely related labels, a challenge compounded by the differences in language-specific text representations.

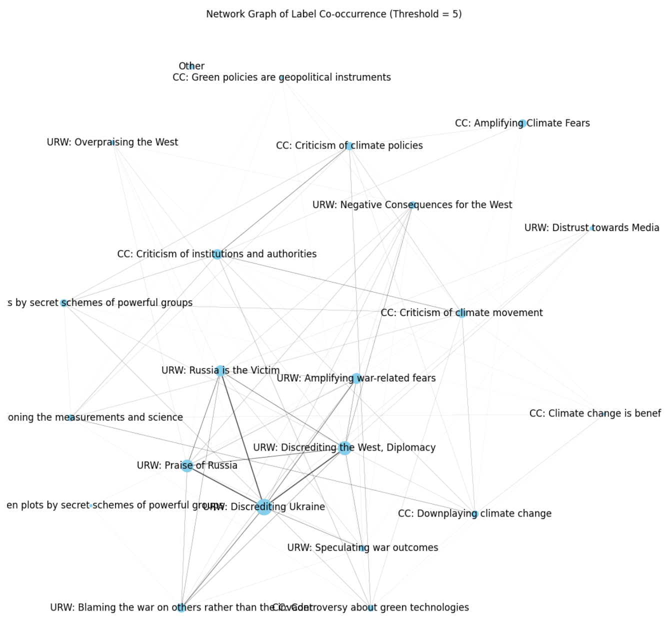
Network Graph of Label Co-occurrence (Threshold = 5)
Other
CC: Green policies are geopolitical instruments
CC: Amplifying Climate Fears
URW: Overpraising the West
CC: Criticism of climate policies
URW: Negative Consequences for the West
URW: Distrust towards Media
CC: Criticism of institutions and authorities
s by secret schemes of powerful groups
CC: Criticism of climate movement
URW: Russia is the Victim
URW: Amplifying war-related fears
oning the measurements and science
CC: Climate change is benef
URW: Discrediting the West, Diplomacy
URW: Praise of Russia
en plots by secret schemes of powerful groups
URW: Discrediting Ukraine
CC: Downplaying climate change
URW: Speculating war outcomes
URW: Blaming the war on others rather than the
CC: Controversy about green technologies

Figure 2: Connection of narratives

On the other hand Portuguese texts form clear clusters in the embedding space. This suggests stronger language or theme separations. As a result, it might be easier to spot subnarratives in Portuguese content using the same classification methods. So, the dataset offers a tough but useful comparison. It shows how differences in text structure and word choice across languages can affect how well subnarratives are located[1].

1. **Lexical and Diversity Measures:** To assess the linguistic richness of the dataset, we calculated the **Type-Token Ratio (TTR)** [12], which measures the ratio of unique words (types) to the total number of words (tokens) in the corpus.

$$TTR_{URW} = \frac{7864}{65746} = 0.120$$

$$TTR_{CC} = \frac{6087}{33414} = 0.182$$

2. **Word Frequency Distribution:** This will allow us to understand the frequency of words across the entire dataset, highlighting the most common terms that appear in the articles.

3. **Syntactic and Grammatical Measures:** We will perform **Part-of-Speech (POS) tagging** [13] to analyze the frequency and proportions of different parts of speech, such as nouns, verbs, adjectives, and adverbs. This will help us understand the grammatical structure and identify patterns within the text.

Preprocessing

The preprocessing pipeline consisted of seven steps, including an additional step to address class imbalances.

- **Lowercasing:** All letters were converted to lowercase to ensure uniformity.
- **Expansion of Contractions:** Common contractions were expanded (e.g., *isn't* $\rightarrow$ *is not*) to simplify subsequent processing.
- **Removal of Numbers and Non-ASCII Characters:** Non-textual elements were removed to focus on meaningful content.
- **Tokenization:** The text was split into individual tokens (words or subwords) [14, 15].
- **Lemmatization:** Words were reduced to their base forms (e.g., *running* $\rightarrow$ *run*) [16].
- **Removal of Stop Words, Punctuation, and Single-Letter Words:** Uninformative words and symbols were eliminated to refine the text [17].
- **Handling Class Imbalance:** Oversampling was used to increase the presence of underrepresented classes by duplicating corresponding samples, ensuring a more balanced dataset for training [18].

Another important level of data preprocessing was the normal cleaning of raw text in a somewhat systematic manner. This entailed removing irrelevant information such as HTML tags, URLs, special characters, and multiple punctuation without losing the linguistic features relevant for classification. Highly frequent stopwords were also reduced; however, a small selection of function words, which may carry relevant information-for example, negations like not-was retained so as not to lose too much semantic information. This cleaning phase also included normalizing text by converting all words to lowercase, thereby avoiding unintentional duplications of the same token in different capitalized forms [19].

By following these steps in cleaning, the average number of words per article went down from about 400 to 200. This two-fold reduction was quite helpful for models like BERT, which have strict limits on token lengths-512 tokens in this case. This process of removing redundant or noisy data gave the model a clearer concentration on meaningful discourse features and similarly decreased the likelihood of truncating important information in longer articles. Figure 3 shows the distribution of word count before and after the text cleansing, indicating how the dataset was condensed.

Beyond keeping sequences within permissible token boundaries, text cleansing helped to reduce potential model confusion from random symbols and irrelevant fragments. Such "noise" can impede the learning process by introducing unnecessary entropy in the embedding space. With fewer words in total, the dataset also became more homogeneous in length distribution, simplifying batch construction and thus potentially speeding up training convergence. This was particularly relevant in a multi-label classification context, where excessive input length variability can compound computational overhead.

This heavy drop in average word count ultimately yielded a leaner, more semantically coherent corpus to train on. While aggressive data cleaning can sometimes result in the loss of some contextual elements, the resulting gains in classifier performance and computational efficiency generally outweigh that drawback. Thus, the post-processed dataset was well-aligned with the 512-token limit, allowing nuanced classification of subnarratives without discarding critical information from each article.

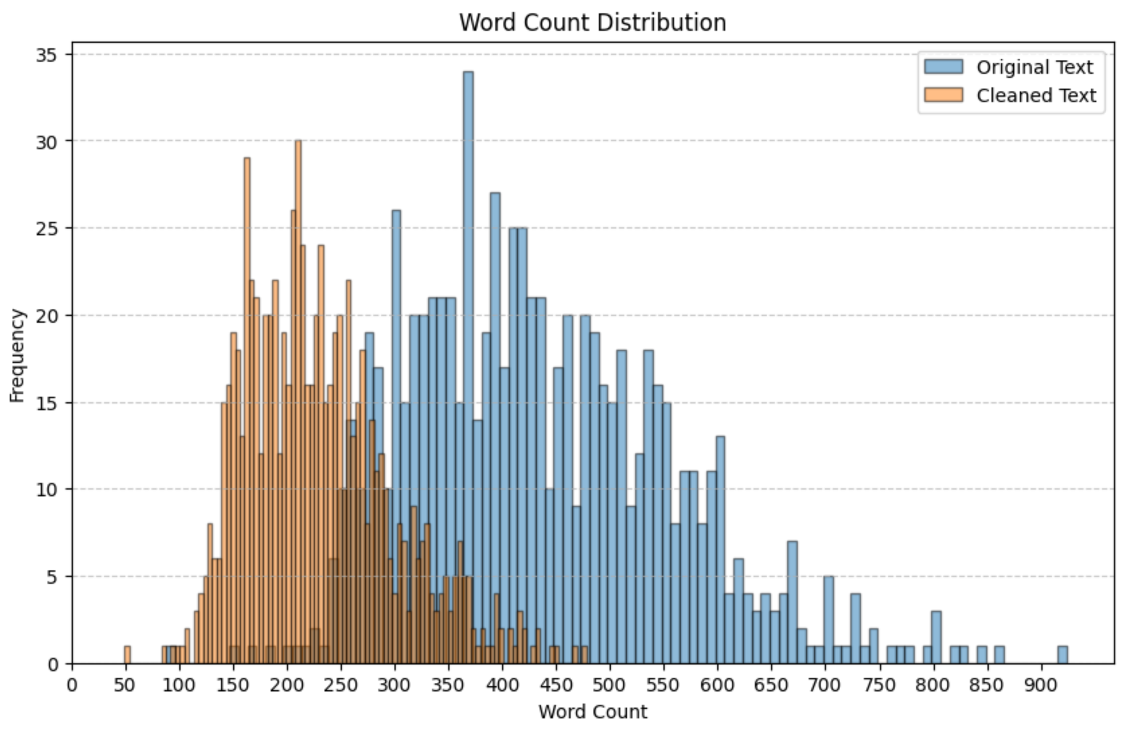

Figure 3: Text Processing Impact on Word Count Distribution

Label balancing across classes and subnarratives was an important step in preparing the data for classification. This task operates under a multiclass, multilabel setup-meaning each article may carry more than one label-so some subnarratives were grossly underrepresented compared to others. Balancing here consisted of a strategy involving oversampling the minority classes by duplicating or synthesizing additional instances. In practice, this meant that some labels had to be duplicated two or three times, while severely underrepresented subnarratives were sometimes multiplied by ten.

While such duplication results in a distribution that can appear contrived-especially when it is considered in isolation-it is quite effective against the risk of a classification model completely ignoring the uncommon labels. It enables the model to see enough examples from each subnarrative that it significantly improves its generalization into the real world, where such less frequent narratives would still occur even if their rate is low. This needs to be done with care, as repeated replication of minority classes runs the risk of overfitting-a model may become too familiar with specific samples, especially when the bounds of typical oversampling are exceeded.

Already the multiclass and multilabel nature further complicates balancing procedures. In quite a few instances, articles carrying several labels were oversampled just like articles bearing only one label. While it is certainly assured that each label in itself does become more visible, this could inadvertently lead to an uneven distribution of the makeup of the articles, that some label combinations started increasing disproportionately, if they would happen to coincide with several minority labels. These nuances notwithstanding, the net effect is salutary: it allows the classification models to make finer distinctions among subnarratives that would otherwise be lost in extremely imbalanced classes.

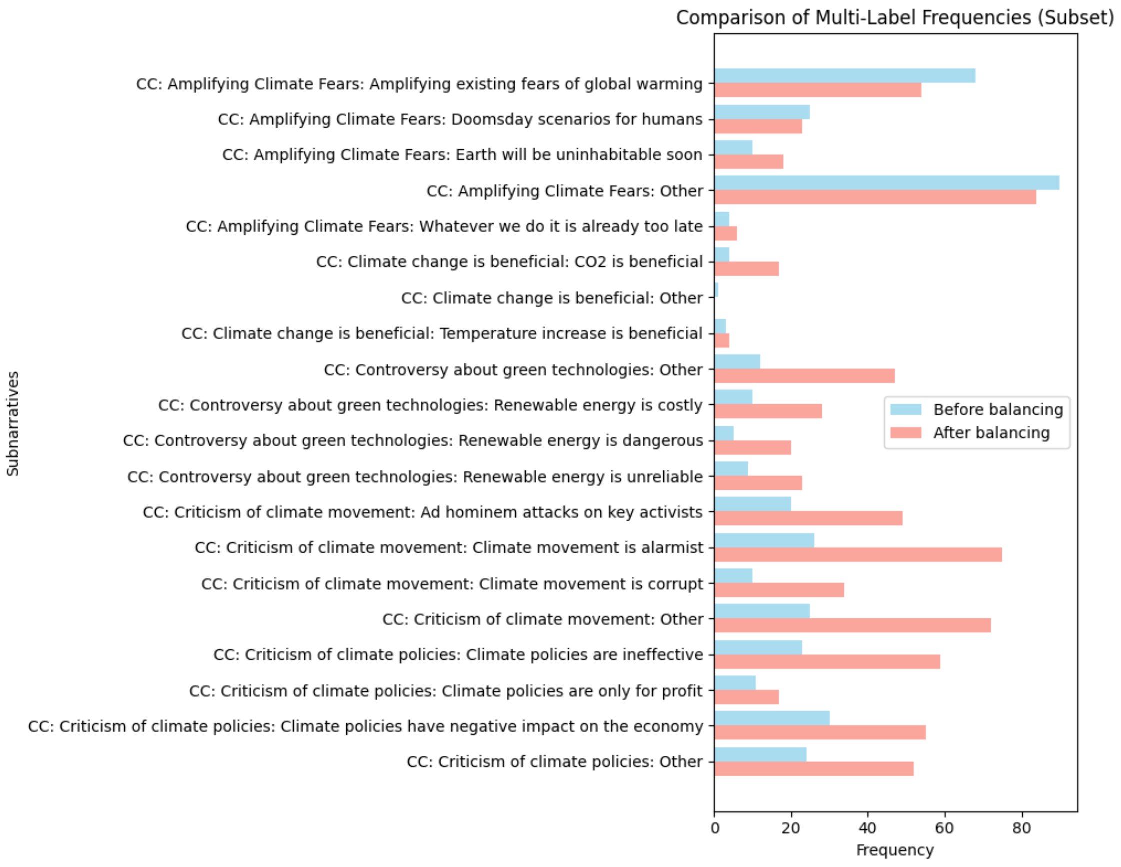

Figure 4: Differences between before and after balancing the subnarrative classes

Figure 4 presents the impact of balancing procedures by comparing the original distribution of subnarratives with the distribution after oversampling. It is easy to see from this side-by-side comparison how previously rare labels become much more prominent once the dataset has been reweighted or augmented. The resulting distribution may be very different from an organically occurring frequency profile, but this shift is both intentional and necessary for the classifier to achieve higher reliability, especially in applications where the accurate detection of rare subnarratives is crucial.

Balancing is a trade-off strategy that tries to enhance the representativeness of minority subnarratives at the possible cost of introducing repeated samples. The increased coverage of less frequent classes helps machine learning algorithms to build stronger decision boundaries, which overall perform better in classifying subnarratives of a complex real-world setting.

4. Methodology:

This section outlines the techniques, algorithms, and models employed in the project, encompassing traditional machine learning, deep learning, and transformer-based approaches. The architectures of custom models are described, along with the pre-trained models utilized.

Naive Bayes

Naive Bayes is a probabilistic classification algorithm based on Bayes' theorem, which describes the relationship between conditional probabilities [20]. Given a set of features, the classifier predicts the probability that a data instance belongs to a particular class. The basis of Naive Bayes is Bayes' Theorem, expressed as

$$P(C|X) = \frac{P(X|C)P(C)}{P(X)} \tag{1}$$

where $P(C|X)$ is the posterior probability of the class C given the observed features X, $P(X|C)$ is the likelihood of the features given the class, $P(C)$ is the prior probability of the class, and $P(X)$ is the marginal probability of the features across all possible classes [21]. The Naive Bayes assumption simplifies the calculation by assuming conditional independence between features, meaning that each feature contributes independently to the probability of a given clas[20]. Mathematically, this means that

$$P(X|C) = P(x_1|C)P(x_2|C)\dots P(x_n|C) = \prod_{i=1}^{n} P(x_i|C) \tag{2}$$

where $x_1, x_2, \dots, x_n$ represent individual features. Despite its simplicity, this assumption often holds sufficiently well in practice, allowing Naive Bayes to be highly effective in various applications, including text classification, spam filtering, and medical diagnosis [21]. The classifier assigns a data point to the class with the highest posterior probability:

$$C^* = \arg\max_{C} P(C) \prod_{i=1}^{n} P(x_i|C) \tag{3}$$

There are several variants of Naive Bayes, depending on how the feature probabilities are estimated. The most common types include Gaussian Naive Bayes, which assumes a normal distribution for continuous features [20], Multinomial Naive Bayes, suitable for text classification where the features represent word frequencies[22], and Bernoulli Naive Bayes, which models binary features [20].

Despite its computational efficiency and strong theoretical foundation, Naive Bayes has limitations, particularly its reliance on the independence assumption, which may not always hold in real-world data[21].

The implementation of Naive Bayes classification in this work relies on Scikit-learn (sklearn) [23] to facilitate model training. The CountVectorizer module from sklearn.feature_extraction.text was used to convert textual data into numerical feature vectors. The MultiLabelBinarizer and LabelEncoder modules from sklearn.preprocessing were used to encode categorical and multi-label data into numerical formats. The classification model was implemented using OneVsRestClassifier from sklearn.multiclass and the MultinomialNB module from sklearn.naive_bayes.

Random Forest

Random Forest is an ensemble learning method that constructs multiple decision trees during training and combines their predictions to improve classification accuracy. It is particularly useful for handling noisy data and reducing overfitting compared to single decision trees [24].

$$C^* = \arg\max_C \sum_{t=1}^{T} P(C|X, T_t) \tag{4}$$

where T_t represents individual decision trees in the ensemble, and $P(C|X, T_t)$ is the probability of class C given the input X for tree T_t. The final prediction is made based on the majority vote across all trees.

- **Vectorization Method:** TF-IDF with n-gram range (1,3), max features = 10,000.
- **Hyperparameters:**
  - Number of estimators (trees): 100
  - Maximum depth: None (unlimited depth)
  - Bootstrap sampling: Enabled
  - Criterion: Gini impurity
  - Random state: 42 (for reproducibility)
- **Feature Selection:** No explicit dimensionality reduction, relying on TF-IDF weights.

To address the issue of class imbalance in the dataset, we applied class weighting strategies. Specifically, we assigned higher weights to underrepresented classes using the "balanced" option in the RandomForestClassifier. This approach ensures that the model does not favor majority classes disproportionately.

Classical NN

The architecture we used is made up of a pre-trained Word2Vec embedding layer, succeeded by a classical Multi Layer Perceptron (MLP). The MLP is a feed-forward neural network with a layered architecture and fully connected neurons [25], meaning that each neuron (or "node") in one layer can receive input from every neuron in the previous layer and can pass its output to every neuron in the subsequent layer. This layout is extended by several dropout layers to combat vanishing gradients, as well as a pooling layer and the addition of language information following layer four. Further details on the implementation are described below. Prior to training, the dataset was adjusted to diminish class imbalances.

Implementation

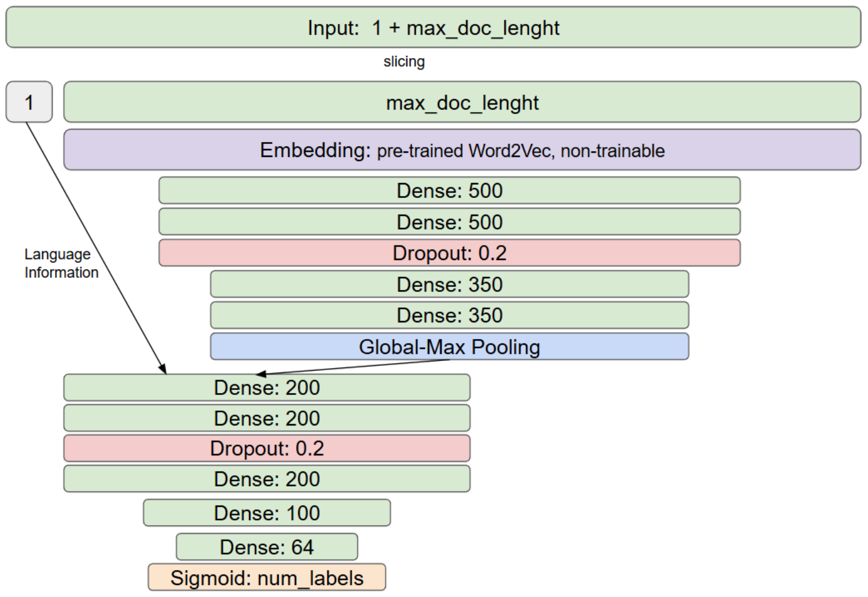

Figure 5: Architecture of the MLP-like neural network approach.

The model described by Figure 5 features a total of 2.1 million parameters, 782.000 of which are trainable. Note that this architecture is used twice, to predict the coarse labels (”narratives”) and to predict the fine labels (”sub-narrtives”), doubling the parameter count for the final model. Regarding the language information, either a 1 or -1 is passed down depending on the text's language (english/portugese). The embedding layer is trained prior to the actual training process of the MLP and is configured to an embedding dimension of 100 and a window size of 12. Most layers use ”relu” as their activation function, only the one preceding the sigmoid-output layer uses ”elu”. The model employs the ”Adam”-Optimizer to minimize ”binary_crossentropy” over 50 epochs.

LSTM

LSTMs use a specialized memory mechanism to regulate long-term dependencies through a dedicated cell state. The LSTM cell regulates the long-term memory (cell state) to enable stable information flow during training. It consists of three gates: the forget gate, input gate, and output gate, each controlling the cell state c_t and hidden state h_t.[26]

Forget Gate: The forget gate decides which information to discard from the previous cell state c_{t-1}. It uses the sigmoid function to output values between 0 (forget) and 1 (keep)[26], modifying the cell state as follows:

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

$$c_f = c_{t-1} \times f_t$$

Input Gate: The input gate determines how to update the cell state. It uses two functional units: one with a tanh activation to propose the change $\tilde{c_t}$, and another with a sigmoid activation to control the magnitude m_t.[26] These are combined and added to the cell state:

$$\tilde{c_t} = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$
$$m_t = \sigma(W_m \cdot [h_{t-1}, x_t] + b_m)$$
$$c_t = c_f + \tilde{c_t} \times m_t$$

Output Gate: The output gate uses a trained matrix W_o and bias b_o to compute relevant information from the current input and previous output. This is combined with the updated cell state c_t to produce the next output h_t[26]:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \times \tanh(c_t)$$

Implementation: The tested LSTM implementation uses word embeddings, represented as a trainable lookup table of dense vectors, where each vector corresponds to a word in the corpus. Initially randomized, these embeddings are optimized during training to capture meaningful features relevant to the task of narrative classification. The output of the LSTM is fully connected to three trainable linear layers, which are used to predict the language, the main and sub-narrative labels.

For loss calculation, cross-entropy is used for the language classification, while binary cross-entropy is ysed to the main and sub-narrative labels, as these labels can be viewed as independent of each other. These loss calculations are then used to optimize the parameters of both the lookup table and the LSTM using the Adam optimizer.

The entire pipeline was implemented in Python using PyTorch[27].

Class imbalances: The binary cross-entropy loss function in PyTorch allows for integrating label-specific weights as a method to address class imbalances in both main and sub-narrative labels[27]. Class weights are computed by summing label occurrences and normalizing them by the maximum class sum. These weights are then applied in `BCEWithLogitsLoss` to assign higher importance to underrepresented labels from both main narrative and sub-narrative classes.

Bert

The following method applies two transformer-based models, BERT and Llama3.1-8B, to classify subnarratives in both English and Portuguese texts related to URW and CC. The classification task operates under a multi-label framework—to reflect the reality that articles often contain overlapping subnarratives. Both models are trained and fine-tuned on data featuring varying levels of class representation, with particular emphasis on managing label imbalance and the potential introduction of new, unintended subnarratives. Fine-Tuning BERT [28] is initialized with a maximum input size of 512 tokens and fine-tuned over 100 epochs. The optimizer of choice is AdamW, using a learning rate of 2×10^{-5}.

Weighted loss is used in order to handle significant class imbalance, especially for less frequent subnarratives. In this way, BERT should be able to provide a better account of underrepresented linguistic markers in both English and Portuguese. Since some of the sub-narratives are nearly specific to one language, multilinguality of BERT captures cross-lingual dependencies by modelling contextual embeddings that remain robust across both English and Portuguese inputs.

Llama

It has been performed fine-tuning on Llama3.1-8B with an instruction-based paradigm, while supporting a token length of as many as 2048 to support longer sentences. In fine-tuning, 100 steps are performed using AdamW but at a learning rate of 2×10^{-4}.

This approach uses a Focal Loss that focuses on hard-to-classify subnarratives and dampens the influence of well-represented examples to handle class imbalance [29]. However, because Llama3.1-8B is capable of free-form text generation, it sometimes introduces novel labels or drifts away from the predefined set of subnarratives. To address this, the instruction templates explicitly limit valid output classes, and any outputs not matching these classes are subjected to post-processing checks [30].

English and Portuguese often exhibit differing subnarrative distributions, necessitating careful handling of bilingual input. Llama3.1-8B's broader context window allows the model to capture longer dependency chains—especially relevant when Portuguese text contains extensive qualifiers or when English articles present complex nested clauses. Despite these advantages, the introduction of new classes by Llama3.1-8B remains a recurring challenge. This addresses that concern through iterative refinements in instruction prompts and additional filtering steps to maintain adherence to the recognized subnarrative taxonomy.

Meanwhile, Llama's more constrained token limit and weighted loss approach provide a complementary perspective that balances computational efficiency with multilingual capacity.

5. Experimental Setup

This section describes the experimental setup, including data preprocessing, model training, hyperparameter tuning, evaluation metrics, and error analysis, to systematically compare the performance of the different models.

Workflow

The workflow starts with data collection and preprocessing. The raw text undergoes cleaning steps like the removal of irrelevant tokens, such as URLs and special characters, and handling of stopwords. After that, balancing is done to avoid class imbalance, ensuring that both frequent and underrepresented subnarratives are well represented. This step is particularly important in multi-label scenarios where one article can have multiple overlapping classes.

Following preprocessing, feature extraction is done using methods ranging from traditional TF-IDF to advanced embedding models. Once the features are prepared, various machine learning models are trained, including Naive Bayes, Random Forests, Neural Networks, and RNNs. Besides that, transformer-based models such as BERT, Llama will be fine-tuned on the same dataset in order to measure the performance gain. The performance of all models is evaluated by using metrics such as F_1-Scores, precision, and recall for an overall comparison of their effectiveness in identifying nuanced subnarratives from within the media articles. This pipeline shown in this Figure 6.

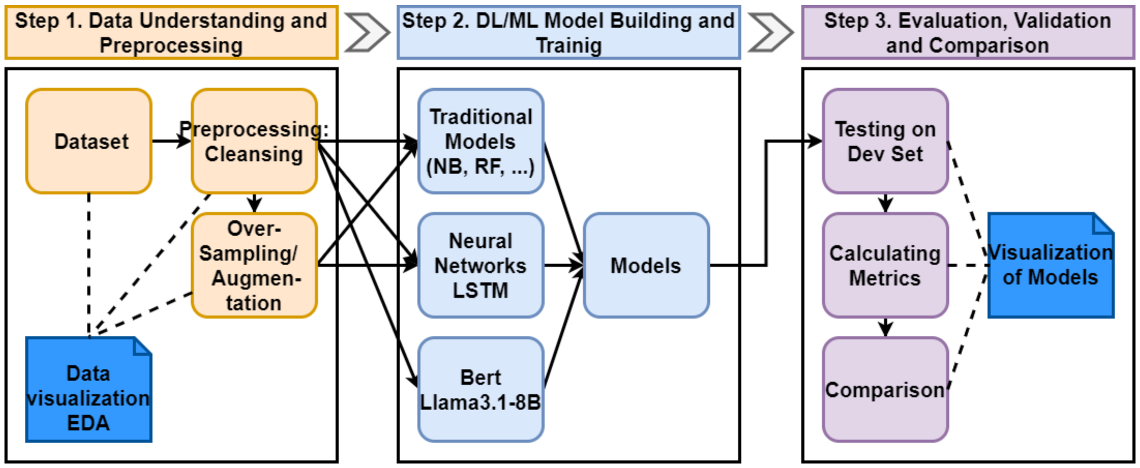
Step 1. Data Understanding and Preprocessing
Step 2. DL/ML Model Building and Trainig
Step 3. Evaluation, Validation and Comparison
Dataset
Preprocessing: Cleansing
Over-Sampling/ Augmen-tation
Data visualization EDA
Traditional Models (NB, RF, ...)
Neural Networks LSTM
Bert Llama3.1-8B
Models
Testing on Dev Set
Calculating Metrics
Comparison
Visualization of Models

Figure 6: Architecture of methods and techniques.

Evaluation Metrics

In the evaluation of models for narrative detection, it is crucial to assess the performance of the classification system using various metrics that provide insights into its ability to correctly identify narratives. Below, we outline the key metrics and strategies for maximizing performance.

The following metrics will be employed to evaluate the models:

- **Accuracy**: The proportion of correctly classified instances out of the total number of instances [31].
- **Precision**: The proportion of correctly classified positive instances out of the total instances predicted as positive. It indicates how many of the predicted positive instances are actually positive [32].
- **Recall**: The proportion of correctly classified positive instances out of the total actual positive instances. It indicates how many of the actual positive instances are correctly identified [33].
- F_1**-Score**: The harmonic mean of precision and recall [34].

Setup

The experimental setup consists of multiple stages, including model training, hyperparameter tuning, evaluation, and error analysis.

Initially, each model is trained independently, and hyperparameter tuning is conducted to identify the best performing set of parameters for each model. The optimal hyperparameters are selected based on their impact on classification performance. Once the best configurations are determined, we compare the highest-performing versions of each model.

Model evaluation is conducted primarily using the F_1-Score, as it provides a balanced measure of precision and recall, which is crucial for assessing classification performance. We analyze model results both before and after data preprocessing techniques, such as data cleaning and oversampling, to determine their influence on classification outcomes. The impact of these preprocessing steps is examined by comparing model performance under different conditions.

Overall, this experimental setup ensures a comprehensive comparison of different models, enabling an informed evaluation of their strengths and weaknesses in narrative classification.

6. Evaluation and Results

This section presents the classification results for each model and compares their performance based on key evaluation metrics such as F_1-Score, precision, and recall. By analyzing these results, we highlight the strengths and weaknesses of different approaches, providing insights into their effectiveness for narrative detection.

Naive Bayes

For Naive Bayes, alpha values between 0.1 and 2.0 were tested, along with both prior fitting (where class probabilities are learned from data) and no prior fitting (assuming equal class probabilities). Alpha is a smoothing parameter used in Laplace Smoothing to prevent zero probabilities for unseen features, ensuring better generalization. Additionally, different n-gram sizes were evaluated, and results were compared with and without oversampling. As visible in Figure 3 for narrative classification, the best hyperparameters varied depending on whether the dataset was balanced through oversampling or left as is. The balanced dataset achieved the highest F1 score of 0.51 with an alpha of 1.00 and an n-gram range of (1,2), yielding a precision of 0.54 and a recall of 0.48. In contrast, the unbalanced training dataset performed best with an n-gram range of (1,1) and an alpha of 1.00, achieving an F1 score of 0.51, precision of 0.48, and recall of 0.53. These moderate F1 scores indicate that while the model is capable of distinguishing narratives, there is room for further improvement. For subnarrative classification, the results also varied based on dataset balancing. The balanced dataset performed best with an alpha of 0.75 and an n-gram range of (1,2), achieving an F1 score of 0.35, precision of 0.46, and recall of 0.29. The unbalanced training dataset, on the other hand, yielded the highest performance with an alpha of 1.00 and an n-gram range of (1,1), achieving an F1 score of 0.35, precision of 0.39, and recall of 0.32.

Table 2: Performance Metrics for Naive Bayes

	Narrative Classification		**Subnarrative Classification**	
Metric	No Oversampling	After Oversampling	No Oversampling	After Oversampling
Accuracy (%)	26	25	16	16
Precision (%)	48	54	39	45
Recall (%)	53	48	32	29
F_1-Score (%)	50	**51**	35	**35**

Random Forest

Random Forest achieved only minor improvement when oversampling technology was applied because the overall accuracy rose from 31% to 32% for narratives and from 15% to 17% for subnarratives. Random Forest achieved its best results in narrative classification after applying oversampling by reaching an F_1-Score of 51% while the baseline score was 14%. Oversampling increased the subnarrative classification F_1-Score to 35% while generating slight precision improvement from 39% to 45% and leaving the score at 35%.

Table 3: Performance Metrics for the Random Forest

Metric	No Oversampling	After Oversampling	No Oversampling	After Oversampling
Accuracy (%)	31	32	17	15
Precision (%)	19	57	5	31
Recall (%)	14	24	2	9
F_1-Score (%)	14	**34**	3	**11**

Classical NN

The Classical Neural Network showed an inconsistent reaction to sample oversampling implementation. Oversampling led to a decline in narrative accuracy from 22% to 17% yet resulted in substantial recall improvement from 42% to 58% which created a mild increase of the F_1-Score between 47% and 50%. The subnarrative classification demonstrated decreasing results from accuracy and precision with accuracy dropping to 8% and precision reaching 32% whereas recall stayed at 28% to yield a F_1-Score drop from 35% to 30%.

Table 4: Performance Metrics for the classical NN

	Narrative Classification		**Subnarrative Classification**	
Metric	No Oversampling	After Oversampling	No Oversampling	After Oversampling
Accuracy (%)	22	17	17	8
Precision (%)	54	43	47	32
Recall (%)	42	58	28	28
F_1-Score (%)	47	**50**	**35**	30

LSTM

Applying oversampling (OS) and weighted classes (WC) improved the LSTM's performance, particularly in subnarrative classification.

In narrative classification, accuracy increased from 17% to 23%. Precision slightly dropped from 50% to 41%, while recall improved from 19% to 21%, maintaining an F_1-Score of 28%. For subnarrative classification, accuracy improved from 7% to 13%, precision increased from 30% to 36%, recall rose from 6% to 10%, and the F_1-Score improved from 11% to 16%. These results were achieved by an LSTM with 1024 units and a randomly initialized 1024-dimensional embeddings layer trained together.

Table 5: Performance Metrics for the LSTM

Metric	**Narrative Classification**		**Subnarrative Classification**	
	No Oversampling	After Oversampling	No Oversampling	After Oversampling
Accuracy (%)	17	23	7	13
Precision (%)	50	41	30	36
Recall (%)	19	21	6	10
F_1-Score (%)	28	**28**	11	**16**

BERT

BERT showed robust performance in both narrative and subnarrative classification. For narrative classification, it achieved a 31% accuracy, with a precision of 65%, recall of 48%, and an F_1-Score of 55%. In the subnarrative task, BERT reached 21% accuracy, accompanied by a precision of 45%, recall of 33%, and an F_1-Score of 38%.

Table 6: Performance Metrics for BERT

Metric	**Narrative Classification**	**Subnarrative Classification**
Accuracy (%)	31	21
Precision (%)	65	45
Recall (%)	48	33
F_1-Score (%)	55	38

Llama

Llama also showed modest performance in both classification tasks. For narrative classification, it achieved an accuracy of 27%, 57% precision, 48% recall, and F_1-Score of 52%. In subnarrative classification, the model achieved an accuracy of 11%, 31% precision, 31% recall, and an F_1-Score of 31%.

Table 7: Performance Metrics for Llama

Metric	**Narrative Classification**	**Subnarrative Classification**
Accuracy (%)	27	11
Precision (%)	57	31
Recall (%)	48	31
F_1-Score (%)	52	31

Overall Results

Table 8 presents the F_1-Scores for Naive Bayes, Random Forest, Classical Neural Network (NN), Long Short-Term Memory (LSTM), BERT, and LLama. For Narrative Classification, BERT achieves the highest F_1-Score of 0.55, followed by LLama at 0.52 and Naive Bayes at 0.51. Classical NN performs similarly to Naive Bayes with a score of 0.50, while LSTM scores 0.42, and Random

Forest has the lowest score at 0.34. In Subnarrative Classification, BERT again achieves the highest score at 0.38, followed by Naive Bayes at 0.35. LLama and Classical NN show similar performance with scores of 0.31 and 0.30, respectively. LSTM performs slightly worse at 0.27, while Random Forest has the lowest score at 0.11.

Overall, BERT consistently outperforms the other models in both classification tasks. Naive Bayes also performs relatively well, particularly in Subnarrative Classification. Random Forest exhibits the weakest performance across both tasks.

Table 8: F_1-Score for all explored models, before and after the application of oversampling, for both the coarse and fine classification tasks. The best scores are emboldened.

Model	**Narrative Classification**	**Subnarrative Classification**
Naive Bayes	0.51	0.35
Random Forest	0.34	0.11
Classical NN	0.50	0.30
LSTM	0.42	0.27
BERT	**0.55**	**0.38**
LLama	0.52	0.31

7. Analysis and Discussion

This section examines the key findings from the results, evaluating the effectiveness of different approaches while identifying their strengths and limitations. We explore unexpected observations, address challenges encountered during the project, and consider potential improvements.

Naive Bayes

The results indicate that dataset balancing, does not consistently improve performance. While for narrative classification the balanced dataset performed slightly better, for subnarrative classification, the highest F1 scores were identical across both datasets, with the unbalanced dataset yielding slightly better precision and recall. This suggests that oversampling does not provide a significant advantage and may even introduce noise in more complex classification tasks. The observed differences in precision and recall values highlight trade-offs in false positives and false negatives, guiding potential refinements to improve overall classification performance.

Among the models evaluated, Naive Bayes achieved a relatively strong performance in narrative classification ($F_1 = 0.51$) and subnarrative classification ($F_1 = 0.35$), outperforming some of the more complex models in certain cases. This result can be attributed to Naive Bayes' ability to handle high-dimensional sparse feature spaces efficiently, making it well-suited for text classification tasks. Since Naive Bayes operates under the conditional independence assumption, it tends to work well when individual words contribute independently to classification decisions, as is often the case in bag-of-words or TF-IDF-based representations. However, its assumption of feature independence can also be a limitation, especially for subnarrative classification, where contextual dependencies between words may play a larger role. This limitation may explain why more complex models like BERT and LLaMA outperformed Naive Bayes, as these models can capture deeper semantic relationships within the text.

Random Forrest

Random Forest, while generally a robust classifier, demonstrated relatively poor performance in both narrative and subnarrative classification, primarily due to the small and imbalanced dataset. Decision tree-based models struggle with limited data, leading to high variance and poor generalization. Class imbalance further skews the model's ability to identify minority class instances, lowering recall [24].

Oversampling was applied to address this issue, improving narrative classification from an F1-score of 14% to 34%, while subnarrative classification saw only a minor increase from 3% to 11%. This suggests that oversampling did not sufficiently enhance learning. The model exhibited high precision (rising from 19% to 57%) but low recall (14% to 24%), indicating confidence in predictions but poor coverage of relevant instances.

Overall, the model's is limited by the dataset constraints and its reliance on feature splits. Alternative strategies, such as weighted decision trees, cost-sensitive learning, or SMOTE-based augmentation, could improve performance. Given the superior results of models like BERT and Naive Bayes, future work could explore hybrid approaches integrating multiple classifiers.

NN and LSTM

Since the LSTM processes its input sequentially, one would expect the model to outperform its NN counterpart on this text-related task. However, another key difference between training the NN and LSTM models lies in the word embeddings layer. While the NN relies on a pretrained word embeddings layer, the LSTM trains its embeddings alongside the model. This suggests that pretrained embeddings may be more suitable for the task, though further testing is needed.

The LSTM results seen in 5 indicate that data cleaning improves performance, while oversampling combined with class weighting yields even better results. Class balancing methods were tested separately, but the best outcomes emerged when applied to the cleaned dataset. However, the best F1 scores as seen in 8 were achieved using an LSTM with 512 units and a 512-dimensional embeddings layer, trained on cleaned text without any class balancing method.

Bert and Llama

A direct comparison of BERT and Llama shows that BERT has higher scores in both the subnarrative and narrative classification tasks consistently. To be precise, in subnarrative classification, BERT has an accuracy of 0.21, precision of 0.45, recall of 0.33, and F_1-Score of 0.38. In contrast, Llama has an accuracy of 0.11, precision of 0.31, recall of 0.31, and F_1-Score of 0.31 as shown in the Table 6, 7. The performance gap is seen both in accuracy and precision, with BERT being more accurate at identifying fewer false positives and more consistent at differentiating between the different subnarratives. In narrative classification, BERT's superiority is again seen with an accuracy of 0.31 and F_1-Score of 0.55, against Llama's accuracy of 0.27 and F_1-Score of 0.52. Here, BERT's higher precision of 0.65 indicates that it is better at not predicting the wrong narrative labels, while its recall of 0.48 matches Llama, which also has a recall of 0.48.

A main advantage of BERT is that its attention mechanisms, learned from huge multilingual data, appear to be particularly appropriate for the task where multiple overlapping labels can be present in every input. By putting emphasis on weighted loss and making the fine-tuning more constrained, BERT seems less likely to "drift" towards producing spurious outputs. Its capacity to deal with imbalanced subnarratives—some of which are very rare—also shows a good ability to pick up minority class signals in both English and Portuguese texts.

Llama's strongest point is its flexibility and ability to deal with longer contexts—particularly useful when subnarratives cover long or complicated paragraphs [35, 30]. Yet, the same generative qualities can lead to label drift unless the model is tightly controlled by carefully designed prompts and post-processing mechanisms. While its recall generally continues to be competitive, its relatively lower precision and accuracy imply that it might predict more incorrect classes in total. In fact, the instruction-based method needs careful monitoring to keep the model in line with the limited label set, a problem BERT doesn't struggle with as much.

A surprising finding is that Llama, even though it is a larger model with a wider context window, does not always beat BERT on precision and F1 scores. This suggests that more prolific training data or more tailored prompt engineering may be needed to bring out Llama's full capabilities for classification tasks—particularly those demanding strict compliance with a set taxonomy of subnarratives and narratives.

The main difficulty faced in this task arises from the multi-label nature of the problem and the extreme class imbalance. Although weighted loss for BERT and Focal Loss for Llama alleviate this to a certain degree, obtaining high recall for the rarest subnarratives is still non-trivial [35]. Additionally, bilingual inputs also make model training and evaluation harder since subnarrative cues in English would not necessarily be present in Portuguese, and vice versa.

On a practical level, the results of this project are easily applicable to real-time media monitoring and content analysis. The robust performance of BERT could be utilized for instance in automated reporting systems or fact-checking pipelines where robust, high-precision detection of labeled subnarratives is required. At the same time, Llama's longer context window might be used in situations in which articles are lengthy and complex, though extra precautions—such as more rigid prompts or systematic post-processing—might be warranted to preserve label integrity. Across the board, the tradeoff between accuracy, recall, and flexibility underscores critical considerations for large-scale deployment of these language models [35, 28].

8. Conclusion

This study evaluated the performance of various machine learning and deep learning models in classifying propaganda narratives in media articles related to the Ukraine-Russia war and climate change. The results indicate that transformer-based models, particularly BERT, significantly outperformed traditional and deep learning approaches in both narrative and subnarrative classification.

BERT achieved the highest overall performance, with an F_1-Score of 55% for narrative classification and 38% for subnarrative classification. This superior performance highlights the effectiveness of transformer-based architectures in capturing contextual dependencies and complex linguistic patterns in propagandistic texts. Llama also demonstrated competitive results, achieving an F_1-Score of 52% for narratives and 31% for subnarratives, suggesting its potential as an alternative transformer-based approach.

Traditional machine learning models, such as Naive Bayes and Random Forest, showed limited effectiveness in distinguishing propaganda narratives. While Naive Bayes achieved moderate success, particularly in subnarrative classification (F_1-Score: 35%), Random Forest struggled, with its subnarrative classification F_1-Score dropping to just 11%. These results suggest that models relying on probabilistic feature distributions and decision trees may not adequately capture the intricate rhetorical strategies used in propaganda.

Deep learning models, including Classical Neural Networks and LSTMs, exhibited mixed performance. The Classical Neural Network achieved an F_1-Score of 50% for narratives but showed inconsistency in response to oversampling. LSTM performed moderately, achieving F_1-Scores of 42% for narratives and 27% for subnarratives. While LSTM is effective in capturing sequential dependencies, it lacks the bidirectional attention mechanisms of transformer models, making it less effective in recognizing nuanced propagandistic cues. Also LSTM and NN were not provided enough data to differ significantly from tradition ML algorithms.

The impact of oversampling varied across models. While it generally improved recall, it sometimes reduced precision, leading to minor gains or even slight declines in accuracy. Notably, Random Forest benefited from oversampling in narrative classification but remained the weakest-performing model in subnarrative classification.

Overall, the application of transformer-based models, particularly BERT, significantly enhanced the accuracy and effectiveness of detecting nuanced propaganda narratives. Unlike traditional machine learning models that rely on statistical word distributions or deep learning models that process text sequentially, transformers leverage contextual embeddings, bidirectional attention, and deep feature representations. This enables them to detect subtle propaganda techniques such as emotional manipulation, misleading framing, and implicit biases.

9. Future Work

We have shown several approaches to classify the narratives of texts regarding climate change and the Ukraine conflict. However, there is still room for fine-tuning. Also, some models would benefit from additional training data. One way to obtain this would be by automatically translating news written in various languages. Additionally, acquiring more data and exploring advanced techniques for handling class imbalance could further improve model performance. These techniques include data augmentation using large language models (LLMs), applying synonymization techniques, and translating foreign news sources to diversify the dataset.

Once the desired accuracy has been achieved for a model, it will be suited for application to previously unknown documents. Potential applications range from building a system to provide narrative labels on news websites to filtering out biased articles in order to obtain a set of "neutral" news coverage.

The application findings have useable benefits for future usage in media monitoring work and fact-checking efforts and disinformation detection systems. Embedding transformer-based models into fact-checking pipelines enables automated narrative classification which flags deceptive news content found in social media and online news. These models demonstrate broad applicability by showing their potential for detecting various sorts of misinformation which extends outside climate change and the Ukraine-Russia war domains.

The class imbalance problem requires attention especially because certain rare subnarratives cannot obtain enough training data. Further investigations must examine innovative methods for data augmentation with active learning approaches to enhance model performance among minority classes. The lack of explainability about transformer models presents itself as a primary challenge because these black box models deliver excellent performance despite their opacity. Interpretable artificial intelligence systems for propaganda detection should be developed to improve the transparency as well as trustworthiness of these systems.

The future of the project has strong potential in multilingual generalization studies. The evaluation of narratives and propaganda strategies across linguistic boundaries becomes possible when using multilingual transformer models for cross-lingual narrative classification which enhances system reliability in diverse linguistic and cultural contexts.

Bibliography

[1] Semeval 2025 task 10: Multilingual characterization and extraction of narratives from online news. https://propaganda.math.unipd.it/semeval2025task10/, 2025. Accessed: 2025-02-10.

[2] John W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.

[3] Y. Zeng, S. Li, and M. Thomson. Understanding the spread of propaganda: ... *Scientific Reports*, 11:1–12, 2021.

[4] R. Li, J. Cohen, and A. Smith. Detecting propaganda devices in textual data. In *CEUR Workshop Proceedings*, volume 3117, pages 115–127, 2022.

[5] Illiberalism Studies Program. Tell us how you really feel: Analyzing pro-kremlin propaganda devices & narratives to identify sentiment implications. Online Resource, 2022.

[6] M. Jones, L. Chen, and G. Roberts. Exploring user perceptions of online messaging strategies and credibility. *PLOS ONE*, 18(1):e0291423, 2023.

[7] Karl Pearson. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[8] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933.

[9] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[10] Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(2):109–125, 1981.

[11] Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, 1991.

[12] Brian Richards. Type/token ratios: What do they really tell us? *Journal of Child Language*, 14(2):201–209, 1987.

[13] Eric Brill. A simple rule-based part of speech tagger. In *Proceedings of the third conference on Applied natural language processing*, pages 152–155. Association for Computational Linguistics, 1992.

[14] Gregory Grefenstette and Pasi Tapanainen. Tokenization: A key concept in natural language processing. In *Proceedings of the 15th conference on Computational linguistics-Volume 4*, pages 1028–1032. Association for Computational Linguistics, 1994.

[15] Rico Sennrich, Barry Haddow, and Alexandra Birch. Tokenization in nlp: An overview. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, 2016.

[16] Hongfang Liu, Travis Christiansen, William A Baumgartner, and Karin Verspoor. Biolemmatizer: A lemmatization tool for morphological processing of biomedical text. *Journal of Biomedical Semantics*, 3(1):3, 2012.

[17] Analytics Vidhya. Text preprocessing in nlp with python codes. *Analytics Vidhya*, 2021.

[18] Jason Brownlee. Random oversampling and undersampling for imbalanced classification. *Machine Learning Mastery*, 2020.

[19] Bernard Bekavac and Marko Kolar. The influence of preprocessing on text classification using a bag-of-words representation. *ArXiv*, 2019.

[20] B. Vikramkumar, Vijaykumar, and Trilochan Tripathy. Bayes and naive bayes classifier. 04 2014.

[21] Hong Chen, Songhua Hu, Rui Hua, and Xiuju Zhao. Improved naive bayes classification algorithm for traffic risk management. *EURASIP Journal on Advances in Signal Processing*, 2021(30), 2021.

[22] Shuo Xu, Yan Li, and Wang Zheng. Bayesian multinomial naïve bayes classifier to text classification. pages 347–352, 05 2017.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[24] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[25] Marius-Constantin Popescu, Valentina E Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. Multilayer perceptron and neural networks, 2009.

[26] Christian Bakke Vennerød, Adrian Kjærran, and Erling Stray Bugge. Long short-term memory rnn, 2021.

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[28] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[29] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

[30] Meta AI Research. Llama 3: Advancing open-source large language models. *Meta AI Technical Report*, 2025. Available online: https://ai.meta.com/research/.

[31] Google Developers. Classification: Accuracy, precision, recall, and f1 score, 2023.

[32] Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval.* ACM Press/Addison-Wesley, 1999.

[33] David M. W. Powers. Evaluation: From precision, recall and f-measure to roc, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 2011.

[34] C. J. van Rijsbergen. *Information Retrieval.* Butterworth-Heinemann, 1979.

[35] Hugo Touvron, Thibaut Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Siddharth Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.