Babeș-Bolyai University of Cluj-Napoca

Faculty of Economic Sciences and Business Administration
Doctoral School of Economic Sciences and Business Administration

PhD Thesis Summary

Contributions to the Development of Linguistic Resources for Processing Romanian Short Texts using Machine Learning

PhD Candidate: Dan-Claudiu NEAGU

Adviser: Prof. PhD. Dorina LAZĂR

Cluj-Napoca, 2025

Contents

Contents of the PhD Thesis

List of own publications

1. Neagu, D.C.; Rus, A.B.; Grec, M.; Boroianu, M.A.; Bogdan, N.; Gal, A. *Towards Sentiment Analysis for Romanian Twitter Content*. Algorithms 15(10), pp. 357, 2022 https://doi.org/10.3390/a15100357

2. Neagu, D.C.; Rus, A.B.; Grec, M.; Boroianu, M.A.; Silaghi, G.C. *Topic Classification for Short Texts*. In International Conference on Information Systems Development, Cham: Springer International Publishing, pp. 207-222, 2023, https://doi.org/10.1007/978-3-031-32418-5_12

3. Neagu, D.C. *BERTweetRO: pre-trained language models for Romanian social media content*. Studia Universitatis Babeș-Bolyai Oeconomica, Volume 70, Issue 1, pp. 83-111, 2025, https://doi.org/10.2478/subboec-2025-0005

Chapter 1

Introduction

Supervising social media platforms becomes an effective way of surveillance as the number of their users has skyrocketed around the world: by February 2025, it is estimated by Statista[1] that around 5.56 billion people have access to the internet and use it on a regular basis, which represents $\approx$ 67.9% of the global population. Of this total, 5.24 billion, or $\approx$ 63.9% are social media users.

Romania has a population of slightly over 19 million but reached a staggering 18 million internet users by the start of 2024, with approximately 17.3 million of them being active on social media. This number represents around 90% of the country's population and is considerably higher than the world average[2]. By analyzing data created and consumed in the Romanian online space, valuable insights about public opinions, trends, and personal interest can be identified and used for scientific or commercial purposes.

Natural language processing (NLP) is a subfield of artificial intelligence concerned with providing computers the ability to process data encoded in natural language using either rule-based, statistical, or machine learning (ML) approaches in order to tackle a wide array of tasks like [34]: speech recognition, text classification, natural language understanding and generation, etc.

Microblogging platforms such as Twitter (rebrabded as "X"), Instagram, Facebook, or TikTok inspire provocative questions as they feature linguistic challenges rarely found in literary texts. Eisenstein [23] refers to these as *bad language*, encompassing emoticons, phrasal abbreviations like *lol*, *smh*, and *ikr*, expressive word lengthening (e.g., *cooool*), or terms written in non-standard forms, including typos, irregular vocabulary, or informal grammar. Various reasons not in the scope of our research cause the presence of bad language in social media content and it dramatically influences the performance of a standard NLP model applied here [51].

Sentiment Analysis (SA) is an established category of NLP, with a lot of research efforts being focused on discovering the ML methods that produce the best models given a particular problem under study. Although textbooks such as [39] or reviews such as [25, 76] present in detail the recommended steps to be adopted specifically for SA or with respect to a given technology applied to NLP in general, a lot of research space is still open in the area of SA, if certain problem-specific conditions occur, such as those induced within microblogging platforms, or handling user input from mobile devices, etc.

Specifically targeting Twitter, performing SA on non-English tweets is seen as challenging, mostly because of the difficulty to gather enough labeled data in the target

[1] `https://www.statista.com/statistics/617136/digital-population-worldwide/`
[2] `https://www.statista.com/topics/7134/social-media-usage-in-romania/`

language [10]. Annotated datasets can be found much easily for popular languages in the world. For example, for English, we can mention BERTweet [51], a large-scale language model trained over a corpus of 850M Tweets, which could be used together with *fairseq* [56] or *transformers* [75] for text categorization tasks, including SA. In France, DEFT challenges conducted between 2014 to 2018 focused on opinion mining and SA from Twitter posts [54] by offering the participating teams access to labeled datasets. In Spain, the TASS[3] workshop, held every year since 2012 at the SEPLN[4] congress supplied a dataset with annotated tweets [19], including Spanish crosslingual variations.

However, little could be found for the less popular languages of the world, such as Romanian. Ciobotaru and Dinu [16] performed emotion detection over a dataset of about 4,000 tweets in Romanian. The texts were manually labeled by them, but the dataset was not made public. Istrati and Ciobotaru [29] collected and manually labeled a dataset with Romanian tweets about brands and created a SA model for usage in brand monitoring. Unfortunately, their manually labeled dataset is also not publicly available. To our knowledge, the recently proposed *LaRoSeDa* dataset [70] is the first and only public dataset dedicated to SA in Romanian.

Part of our personal motivation includes a 3-class sentiment prediction capability ("negative", "neutral", "positive"), tailored for social media-specific content. This makes LaRoSeDa an unsuitable candidate for our work because the sentiment is labeled in a binary fashion ("negative" and "positive") and the texts refer to product reviews collected from online shopping sites, not from social media platforms.

Discovering abstract topics that occur in a collection of texts or documents could be done with either Topic Classification (TC) or Topic Modeling (TM). TM is an unsupervised technique [72] that doesn't require labeled data, while TC is a supervised one, where labeled data is needed for model training.

For this study we chose TC over TM for several reasons. Firstly, TM poses some challenges due to its unsupervised nature and the lack of predefined topics. Unlike TC, where the number of topics is determined by the training data, TM may generate an indefinite number of latent topics, making it harder to interpret, evaluate, and apply in real-life scenarios. Secondly, TM requires human intervention to understand and label the newly generated latent topics which adds complexity and subjectivity. On the other hand, TC is a straight forward approach that associates the documents to already known classes which allows for more concrete insights to be drawn.

When it comes to TC for Romanian we mention Vasile et al. [71] who conducted a study in which a number of traditional models were used to categorize 219 blog posts into 9 distinct topic classes. Sequential Minimal Optimization (SMO) and Complement Naive Bayes (CNB) had the best results with an accuracy rate of 77.8%, k Nearest Neighbors (k-NN) achieved a slightly lower score of 73.3%, and the classic Naive Bayes (NB) model had the worse performance with an accuracy of 68.9%.

Other research on Romanian microblogging content is missing despite the potential social and economic benefits that can be obtained from the use of TC systems. This might be due to the problematic characteristics of social media texts, but with the recent advancements in the ML field more complex models, like Transformers, can be used to address these challenges. Another process that could encourage research in this direction would be the collection and annotation of new large-scale datasets.

By integrating both SA and TC we want to develop a NLP system capable of processing

[3] `http://tass.sepln.org/`

[4] `http://www.sepln.org/workshops/neges2019/`

data extracted from Romanian social media platforms from 2 points of view. The SA component will have the role of predicting the global sentiment polarity of the texts, while the TC component will focus on categorizing the texts into a number of predefined discussion themes. With these functionalities at hand, researchers or private entities can obtain in near real time a better understanding of public opinions and trending topics.

Our primary goal is to address the issue of limited linguistic resources which can be used to train reliable SA or TC models for Romanian social media texts. Automated translation of popular datasets into different languages has been presented in studies such as [7, 6] and suggest that certain models have similar performances regardless of the dataset used for their creation.

As our secondary objective, we want to address the need for a comprehensive comparison between different ML approaches. By training and testing a wide range of ML models we hope to answer some important research questions. Is automatic translation from English to Romanian feasible for sentiment or topic classification? What encoding methods and models to choose? Does hyperparameter optimization improve performance? How do models compare in terms of accuracy and execution speed?

Our third objective refers to the creation of Transformer models designed for Romanian social media texts. In order to do this we first need to find, collect, and curate a dataset containing a substantial number of unlabeled Romanian tweets. To be more specific, we want to create several variants of RoBERTa models from scratch using our custom corpus, and these variants will be referred to as BERTweetRO. After the pre-training process, we'll fine-tune the BERTweetRO variants for SA and TC using translated data and compare the performance of the best variant(s) with Multilingual BERT, classic learners, and the deep learners.

Our final objective is to assess the SA performance of our best models against Sentimetric[5], a commercial solution for Romanian SA. In order to have a fair comparison with practical implication, we want to manually collect a small dataset of real life Romanian tweets and, with the help of human volunteers, label each tweet as either negative, neutral, or positive. We'll instruct each volunteer how the labeling processes should be carried out to ensure that this new dataset can serve as a reliable evaluation benchmark.

By comparing the models in each language separately we'll be able to rank them by performance and by comparing the English models against the Romanian ones, we'll be able to see which models adapt best to translated texts. For TC, we want to highlight the benefits of using this supervised approach as opposed to the more common TM approach. Furthermore, our new BERTweetRO model can advance the field of Romanian language processing by allowing other researchers to fine-tune it on other NLP tasks, or to provide a guide on how to pre-train custom BERT models with limited data.

[5] `sentimetric.ro`

Chapter 2

Literature Review

SA involves applying NLP techniques to measure emotions and subjective content in text, being extensively used in areas such as product reviews, survey analysis, social media monitoring, and healthcare. Its applications range from business intelligence and customer feedback analysis to advancements in medical research [27].

Conceptually, there are 2 main approaches which can be used for classifying texts according to their sentiments. In the knowledge-based approach, words such as "happy", "sad", "afraid", or "bored" are considered to denote affect categories [55]. The positive opinion words are used to express a desired state while the negative ones are used to express an undesired state. Some knowledge bases not only list obvious words, but also assign arbitrary words a probable "affinity" to particular emotions [68]. Opinion word lists are usually manually created but can be extended in a automated fashion with the help of dictionaries by finding synonyms and antonyms [45].

Statistical SA incorporates ML methods, including Bag-of-Words (BoW), Latent Semantic Analysis (LSA), Pointwise Mutual Information for Semantic Orientation, word encoding algorithms, etc. These methods offer superior results as they can handle more complex textual data when compared to knowledge-based approaches but labeled datasets are needed to create the models. Among classical ML algorithms, popular choices [43] are Bernoulli Naive Bayes (NB) [42], Support Vector Machines (SVM) [32], Random Forest (RF) [13] or the Logistic Regression (LR) [50]. For DL, all important variants like the standard Deep Neural Network (DNN) [53], the Convolutional Neural Network (CNN) [31], or the Long Short-Term Memory (LSTM) [59] are reported to perform well for text classification.

Classic ML methods and standard neural networks are usually applied on document-level embeddings like TF-IDF [66] or the modern Doc2Vec [37]. DL networks with CNN or LSTM cells are in general applied on word embeddings such as Word2Vec [44]. TF-IDF leads to high-dimensionality problems, so dimensionality reduction schemes are recommended to improve efficiency. In our work, we'll experiment with these methods, in the search for a suitable combination that fits our needs.

Google proposed Bidirectional Encoder Representations from Transformers BERT [18] as a state-of-the-art pre-trained model for many NLP tasks. Multilingual BERT, also pre-trained for Romanian, is reported to work well in cross-lingual knowledge transfer [62]. However, as practice indicates [40], BERT comes with significant time costs for model training and fine-tuning, even on powerful computers.

Performing SA on social media content is seen as a difficult task [10] because one has to deal with bad language [23]. However, for popular languages like English, Spanish, or

French, plenty of linguistic resources that can be applied to enhance SA for microblogging content exit. For English, we mention BERTweet [51], which was fine-tuned for SA and scored an accuracy of 72% on the SemEval2017-Task4A [65] test set, outperforming its competitors RoBERT and XLM-R. Barbieri et al. [9] reports BERTweet as being the state-of-the-art on the TweetEval[1] benchmark, with a 73% average recall. Pota et al. [63] applied BERT-based models for SA on English and Italian Twitter data, highlighting the value of individualized text preprocessing to uncover hidden information, a point we'll consider in our work.

For Romanian, the SA state-of-the-art for microblogging content is less advanced. In the private sector, Technobium[2] created Sentimetric, a web service dedicated to the SA of Romanian texts, with a free demo available online[3]. To our knowledge, a Romanian microblogging content dataset similar to BERTweet is not yet available.

LaRoSeDa (Large Romanian Sentiment Dataset) appears to be the only publicly available Romanian dataset labeled for SA; containing 15,000 product reviews of which 7,500 are labeled positive and 7,500 negative. Due to its nature, all works using this resource report the performances achieved for SA in a binary fashion. For instance, [22] achieved an F1 score of 54%, while in the work which introduced LaRoSeDa an accuracy of ≈91% is reported as the benchmark [70]. More recently, we acknowledge the Romanian DistilBERT corpus[4] which could be employed for binary SA over standard texts. They [5] reported a state-of-the-art binary classification accuracy of 98% for SA performed on LaRoSeDa. Regarding the multinomial SA of social media texts in Romanian, we could not find any published work in order to set a benchmark with which we can compare.

However, Banea et al. [8] responded positively to the question whether we can "reliably predict sentence-level subjectivity in languages other than English by leveraging on a manually annotated English dataset" by training Naive Bayes classifiers on 6 languages starting from an original English dataset with news articles translated with automatic engines. Thus, this motivates our efforts to use machine translation for obtaining learning datasets for Romanian NLP.

Nowadays, discovering abstract topics that occur in a collection of documents could be done with either Topic Classification (TC) or Topic Modeling (TM). TM is a widely used statistical tool for extracting latent variables from large datasets, being well suited for textual data [72]. Among the most used methods for TM we can mention Probabilistic Latent Semantic Analysis (PSLA) and Latent Dirichlet Allocation (LDA) which state that a document is a mixture of topics, where a topic is considered to convey some semantic meaning by a set of correlated words, typically represented as a distribution of words over the vocabulary. In essence, these conventional topic models reveal topics within a text corpus by implicitly capturing the document-level word co-occurrence patterns [74, 12].

Nevertheless, directly applying these models on short texts will suffer from the severe data sparsity problem, i.e. the sparse word co-occurrence patterns found in individual documents [28]. Some workarounds try to alleviate sparsity by aggregating short texts into longer pseudo-documents, though results vary by dataset [4]. The Biterm Topic Model [15], which models the topic components using unordered word pairs (biterms), often outperforms other approaches on short texts.

The main advantage of TM methods is that they do not require labeled data, thus data

[1] `https://huggingface.co/datasets/tweet_eval`

[2] `https://technobium.com/`

[3] `http://sentimetric.ro/`

[4] `https://github.com/racai-ai/Romanian-DistilBERT`

collection becomes more accessible and could be done in a fully or partially automated manner. Despite its popularity, TM is prone to issues with optimization, noise sensitivity, and instability, which can lead to unreliable results [2]. Some techniques also fail to reflect real-world data relationships [11], often due to strong assumptions regarding key parameters. For example, determining the optimal number of topics is non-trivial, and human intervention is needed to assign relevant labels to the identified topics.

When labeled training data is available, TC can be used for topic identification to address many of TM's limitations. Here, popular ML algorithms treat topic identification as a standard classification task using syntactic and linguistic features. Given a training set $D = X_1, X_2, \ldots, X_N$, where each record X_i (document, paragraph, sentence, or word) is labeled with one of k topic classes, the goal is to train models that generalize from these patterns to predict the topics of unseen texts accurately. TC is more transparent and easier to evaluate, making performance assessment and comparison more straightforward.

Zeng et al. [77] proposed a hybrid approach that combines TM with TC. They first extracted the most relevant latent features with TM and then fed them into supervised ML model like SVM, CNN, and LSTM. For the experiments they used the Twitter dataset released by TREC2011[5], which contains around 15,000 tweets, semi-automatically labeled into 50 topic classes. The highest accuracy of $\approx 9.5\%$ was achieved by CNN and can be considered modest at best. Furthermore, they conclude that the TM component did not improve the learning capabilities of the classifiers in any significant way.

Unlike SA, which focuses on text polarity, TC often involves a large and sometimes overlapping number of classes [25, 39]. To overcome this, some authors [26, 52] use Top-K accuracy instead of the standard one. Rather than classifying a text to just one class, the model will produce the K most probable classes and if the actual label is among them, the text is considered to be correctly classified. In our work, we'll take this into account and report the standard accuracy (i.e. Top-1), as well as the Top-2 and Top-3.

Regarding TC for Romanian texts we can only mention the work of Vasile et al. [71] who evaluated the capabilities of some classic ML models when applied to blog content. The data used in their study was extracted from 219 blogs, each instance being labeled with 1 topic class from a total of 9: "Activism", "Business and Finance", "Art", "Travel", "Gastronomy", "Literature", "Fashion", "Politics", and "Religion and Spirituality". The SMO and Complement NB obtained the best results, both reaching an accuracy of around 77.8%. A slightly lower score of 73.3% was achieved by k-NN while the standard NB model had the worse performance of 68.9%. Important to note that the authors used a very small dataset in their experiments which is problematic because it's unlikely that these results can be reproduced on larger evaluation sets.

We couldn't find any other relevant research works that target Romanian social media content and labeled datasets are also missing, meaning that we'll have to translate a suitable English dataset in order to create the training data needed for our TC experiments.

[5] `http://trec.nist.gov/data/tweets`

Chapter 3

Sentiment Analysis for Romanian Social Media Texts

We searched in a number of online platforms including academic databases and NLP repositories (like Kaggle) but couldn't find a dataset that could match our requirements [49]. Thus, we decided to employ an open-source English dataset, translate it to Romanian using Google Translate[1], and use it as a "surrogate" resource in our experiments.

For this research, we selected the Twitter US Airline Sentiment Tweets dataset[2]. The data was collected in 2015 and each tweet was manually labeled by external contributors with its global sentiment polarity (positive, negative and neutral). It contains around 15,000 tweets, with class distribution as follows: 63% negative, 21% neutral, and 16% positive. Each tweet is also accompanied by the contributor's confidence about the annotated sentiment and each negative tweet includes a reason for the assessment.

The structural, grammatical, and syntactical integrity of any text translated with automated processes is affected. The main metric used in the literature to measure the quality of an automated translator is the Bilingual Evaluation Understudy (BLEU) score [58]. This score ranges from 0 to 100 with higher numbers representing a better translation (100 denoting perfect translation). In [3], general English texts were translated to 50 different languages, using Google Translate, and the BLEU score was used as the evaluation metric. The mean score over all the compared translations was approximately 76. English to Romanian achieved a score of 84, which is considerably above average. The maximum BLEU score of 91 was achieved by English to Portuguese while the minimum of 55 was achieved by English to Hindi. Similar results are also reported in [67], where English to Romanian obtained better than average results. These findings indicate that our translation approach for creating a dataset that can be used to train ML models has high chances of success.

For our experiments, the dataset was split into training and test sets using a standard 75–25% split: ≈11,000 instances for training and ≈3,700 for testing, with similar class distributions in both. Moreover, the English and Romanian train and test data are identical in the sense that they contain the same set of instances.

Next we developed a custom preprocessing module, containing the following steps, applied in this specific order:

1. Extra white space removal (language-independent).

[1] `https://translate.google.com/`

[2] `https://www.kaggle.com/crowdflower/twitter-airline-sentiment`

2. Custom word lemmatization and tokenization (language-dependent).
3. URL identification and removal (language-independent).
4. Emoji identification and replacement (language-independent).
5. Social media mention identification and removal (language-independent).
6. Extra consecutive character removal (language-independent).
7. Abbreviation replacement (language-dependent).
8. Stop-word removal (language-dependent).
9. Lower case capitalization (language-independent).
10. Punctuation mark removal (language-independent).

Language-independent steps can be applied in the same manner in both English and Romanian. In contrast, language-dependent steps implies that specific knowledge of Romanian or English is required.

In step 1, all consecutive white spaces which appear more than two times are removed from the tweets, i.e. "Hello world!" becomes "Hello world!".

In step 2, we used SpaCy[3] for word tokenization and lemmatization due to its high accuracy in both English and Romanian. The input for this step are strings and the generated output is a list of tokens, where each token is either a number, a lemmatised word, or a symbol. We chose lemmatisation over stemming because lemmatisation can correctly identify the intended part of speech and meaning of a words.

We modified the default functionality of the tokenization and lemmatisation offered by SpaCy in order to deal with social media specific text, such as:

- We instructed SpaCy not to lemmatise social media specific tagged words (hashtags and mentions) and not to split the tagged words in the tokenization step. By default, SpaCy would transform an input like "#working" into the following list of tokens ["#", "work"]. Therefore, we ensure that social media specific words are kept intact and hashtag tokens are dealt with properly.

- Negated words are crucial for sentiment analysis [25], thus, we implemented a mechanism that can identify negated words within a sentence and appended them with a special prefix and suffix. For example, the string "not happy" is transformed into the following tokens ["not", "|!|happy|!|"]. Having this functionality, we ensure that the negation elements are not lost after the stop-word removal step, as the stop-word lists usually contain negation words like "no", "not".

In step 3, we identify which tokens are URLs (Uniform Resource Locator) using a complex regular expression pattern with over 400 characters and remove them from the data because URLs are usually non human-readable and do not provide any useful information or insights for text classification.

Step 4 is critical for our microblogging context, as we deal with emojis, a sort of "bad language" which have become extremely popular worldwide in informal text sources. For

[3] `https://spacy.io/`

example, Instagram reported in 2015 that nearly half of the text on their platform contained emojis [4]. Kralj Novak et al. [36] reported that around 4% of tweets contain emojis and their sentiment polarity does not depend on the language. With this respect, they constructed the *Emoji Sentiment Ranking* lexicon[5] containing the 751 most frequently used emojis, each annotated with its sentiment polarity (negative, neutral, or positive).

Therefore, in this step we verify if a token is found in the Unicode's *Full Emoji List*[6] and whether it has an associated sentiment in the Emoji Sentiment Ranking lexicon, mentioned above. If it does, the token will be replaced with its polarity together with a special prefix and suffix. For example, "U0001F642" represents the unicode for "slightly smiling face" and has a "positive" polarity. Thus, it will be transformed into "|^|positive|^|". If a token is a emoji but it wasn't labeled with any associated sentiment in the lexicon or the unicode is not an emoji, then another prefix and suffix will be added to it, for example "U00001D19" is the Unicode for "capital reversed R" and will be transformed into "|*|U00001D19|*|".

In step 5, social media mention tags, referencing other social media entities within the network are identified and removed. In the case of Twitter, we search for the "@" symbol which is used to create external links in tweets. Most of the mentions refer to American airlines companies and we noticed a strong correlation between them and the general expressed sentiment of the tweet. Thus, we decided to remove the mentions because their existence might artificially increase the accuracy of our classifiers (we want to assess the polarity based on the language and not based on a specific named entity).

Step 6 deals with excessive consecutive characters. In microblogging texts, regardless of the language, it is common to emphasize a word by adding additional characters. For example the word "cool" might be written with various numbers of "o"s. To solve this issue the extra consecutive characters which appear more than 3 times for a given token are removed. Therefore, we restrict the appearance of a given word to either its standard form or a single instance of emphasized writing: e.g. "cool" can only appear as "cool" or "coool", the second being the emphasized instance.

In step 7, we replace some abbreviations from the text with their corresponding full description with the help of two lexicons we constructed specifically for this purpose. The English version contains around 400 abbreviations, while the Romanian version contains around 100 abbreviations and was built using the Wikipedia page for Romanian abbreviations[7]. After this process each abbreviation is replaced with the tokens derived from its full description. For example, "brb" will be replaced with a list of 3 tokens: ["be", "right", "back"].

In step 8, stop-words are identified and removed using the stop-word dictionaries offered by SpaCy, for both English and Romanian. Removing stop-words is a common task in text processing, as indicated by [39].

In step 9, all tokens are transformed to a lower case capitalization, thus reducing the number of tokens identified for a given concept.

In step 10, we remove all extra punctuation marks within tokens, with the exception of the tokens tagged with our special prefixes and suffixes.

Table 3.1 presents two sample tweets and their representation after applying all 10

[4]https://instagram-engineering.com/emojineering-part-1-machine-learning-for-emoji-trendsmachine-learning-for-emoji-trends-7f5f9cb979ad

[5]`https://kt.ijs.si/data/Emoji_sentiment_ranking/`

[6]`https://unicode.org/emoji/charts/full-emoji-list.html`

[7]`https://ro.wiktionary.org/wiki/Wik%C8%9Bionar:Abrevieri`

steps of our preprocessing pipeline. The first tweet is in English and the second one is its automated translation in Romanian. "U00001F620" is the Unicode placeholder for the "angry face" emoji.

Language	Raw tweet	Preprocessed tweet
EN	"@united and don't hope for me having a nicer flight next time. REALY getting on my nerves U00001F620 #nothappy...."	["|!|hope|!|", "i", "nice", "flight", "next", "time", "really", "nerve", "|^|negative|^|", "|#|nothappy|#|"]
RO	"@unitate și sa nu sperați să am un zbor mai frumos data viitoare. Devine într-adevăr pe nervii mei U00001F620 #nothappy...."	["|!|sperat|!|", "zbura", "frumos", "data", "viitor", "deveni", "adevăr", "nerv", "|^|negative|^|", "|#|nothappy|#|"]

Table 3.1: Tweet preprocessing example

Key to any NLP task is the document internal representation, i.e., properly selecting the features from the raw text and encoding them as numeric values, so as to keep the representation tractable or to enrich it with some language semantics. This process is mandatory because text analysis algorithms require numerical inputs on which to perform mathematical computations. Textual data in its raw form is also filled with many irrelevant or redundant features which should be handled in this stage because the ML models are not so good at processing them on their own [69].

For our study we selected the most popular approaches as suggested by the NLP literature [25, 39]: TF-IDF, Word2Vec, and Doc2Vec.

We trained one TF-IDF vectorizer on the English training set and another one on its Romanian translation. TF-IDF was applied on the preprocessed tweets and the vocabulary was set to contain the tokens which appear at least 3 times. Therefore, we removed a large number of infrequent tokens or those which may have been erroneously built in the preprocessing step. The trained TF-IDF vectorizers were then applied on the testing sets, after which we noticed that the English vocabulary contained around 3,100 tokens while the Romanian one contained around 4,000 tokens, due to the fact that Romanian is generally more verbose than English.

We used the Gensim library [64] to learn the Word2Vec and Doc2vec embeddings for our data. As in the case of TF-IDF, it was necessary to train 2 separate models, one on the English training set and another on the Romanian training set. For Word2Vec we selected the CBOW architectural model because it works better on short texts. For Doc2Vec we used DBOW with a hierarchical softmax architecture. This combination allows for the prediction of words in their context and improves the training time which is ideal in our case. The vocabulary was also set to contain only the tokens that appear at least 3 times.

For both Word2Vec and Doc2vec we set the embedding size to be 200 to ensure that the models can capture enough contextual information and at the same time to maintain an efficient computational performance. We trained each model with a learning rate (α) of 0.025, a window size of 5, over 5 epochs.

Dimensionality reduction in data science and ML refers to the process by which the number of features or their size is reduced such that the available information retains some meaningful properties of the original data [1]. This approach is helpful for some NLP tasks where the initial features are extremely high dimensional like in the case of TF-IDF which is known to generate large sparse matrices.

With dimensionality reduction in place we want to reduce the size of our data then check how much the predictive and computational performance of the ML models is affected. We selected the following algorithms to test: Principal Component Analysis (PCA) [33], Non-negative Matrix Factorization (NMF) [60], and Latent Semantic Analysis (LSA) [20].

We applied all 3 algorithms on the TF-IDF datasets and reduced the number of features to 500. This means that the reduced representation for English is around 6.2 times smaller and for Romanian about 8 times smaller than the original. We didn't apply dimensionality reduction on the Word2Vec and Doc2Vec data because the desired vector size of 200 was set before extracting the features. Also, by reducing the size of these embeddings there's a high chance of compromising their quality altogether.

As indicated by the literature [25, 35, 39], classic machine learning (ML) algorithms, deep learning (DL) approaches, or novel language models approaches could be applied for inferring SA models. For our experiments we decided to apply the following learning methods:

- Classic ML:
  - Bernoulli Naive Bayes (Bernoulli NB)
  - Support Vector Machine with a linear kernel (Linear SVM)
  - Random Forest (RF)
  - Logistic Regression (LR)
- Deep Learning:
  - Deep Neural Network (DNN)
  - Long Short-Term Memory (LSTM)
  - Convolutional Neural Network (CNN)
- Advanced Language Model:
  - Multilingual BERT

Bernoulli NB, SVM, RF, LR, and DNN will be paired with TF-IDF, TF-IDF with reduced dimension, and Doc2Vec. LSTM and CNN will be applied on the Word2Vec encoding because they can process and are specialized on multidimensional sequential data. In this case we incorporate an extra embedding layer under the input layer in order to map each token from the text with its corresponding Word2Vec representation. Because for Romanian we do not possess a global word embeddings resource like GloVe [61] for English, we opted to learn the Word2Vec representations from scratch for both languages using only the training sets.

We implemented the classic learning algorithms with the help of *Scikit-Learn* library[8], while for the deep learning algorithms we used *Keras*[9].

For BERT, we used the model available on the Hugging Face transformers[10], called with the base multilingual uncased variant. What is different about this model compared to the other classifiers is the fact that it has its own encoding mechanism called Multilingual Tokenizer and doesn't support features in TF-IDF, Word2Vec, or Doc2Vec form. On top of M-BERT we added a hidden dense layer with 75 nodes and ReLU activation function, followed by the standard classification layer with 3 nodes which produces the sentiment class. Adam was the selected optimizer, with a learning rate of 2×10^{-5} and $\epsilon = 10^{-8}$. The loss function was set to Categorical CrossEntropy.

[8] `https://scikit-learn.org/stable/`

[9] `https://keras.io/`

[10] `https://huggingface.co/docs/transformers/en/model_doc/bert`

Evolutionary Algorithms (EAs) are a family of optimization methods inspired by natural selection. They work by iteratively improving a set of candidate solutions using a fitness function as the evaluation metric. Selection, crossover, mutation, and reproduction processes are applied on the candidates (also called "individuals") to evolve them over a number of generations with the final goal of finding the best individuals as measured by the selected fitness function [73]. The Genetic Algorithm (GA) is the most popular and basic type of EA.

We selected this population-based probabilistic method because it can drastically speed up the hyperparameter optimization while producing a good-enough combination of parameter values. There are more complex variations for the GA algorithm, like Thermodynamical GA, but we decided to stick with a standard GA as it's been shown to outperform Bayesian optimization anyway [46]. One more benefit of evolutionary optimization is that it works in all 3 types of search spaces (continuous, discrete, and categorical) regardless of the classifier on which the optimization is performed.

We used Sklearn-Genetic-Opt library[11] for implementing the GA optimization in relation with our selected classifiers. Sklearn-Genetic-Opt makes use of the DEAP framework[12], which supplies many EA variants needed for solving optimization problems.

The GA was designed as following. Given a number N of parameters to optimize, an individual/chromosome is denoted as a vector $(p_1, p_2, \ldots, p_i, \ldots, p_N)$. In this vector, each p_i represents the value selected for the corresponding hyperparameter N_i. A population consisting of 20 individuals is evolved over 40 generations with a crossover probability of 80% in order to combine the characteristics of the individuals and a mutation probability of 10% to introduce variation in the population. Individuals are selected for the next generation using a standard elitist tournament of size 3. Internally, each individual is evaluated using the accuracy metric as the fitness function, computed with 3-fold cross-validation. In general, convergence is seen after 15-20 generations, thus evolving the population over 40 generations is more than enough to guarantee a good parameter selection.

In the case of the classic ML algorithms all the parameters described in the official Sklearn documentation were optimized. In the case of DNN, we considered among the parameters the following: the network capacity (the number of hidden layers and the number of units per layer), the activation function, the regularization function, and dropout rate. For CNN and LSTM, we wanted to use the logic but an unexpected issue occurred. For text classification, these two models need the embedding weight parameter to be in the form of a 2D tensor but Sklearn-genetic-opt doesn't support this. As a result we had to modify the source code of the library in order to transmit the multidimensional parameters directly to DEAP.

For the BERT-based classifier, as learning just one model is very time consuming, we omitted to perform the evolutionary optimization procedure. Instead of cross-validation, we reserved 10% of the training set for validation and we let the learning to optimize the loss function for several epochs. We noticed that the model rapidly overfits, thus, we stopped the training process after 2 epochs for both English and Romanian.

We next present the high-level architecture of our Sentiment Analysis system, summarized in Figure 3.1.

At the top of the diagram, the automatic translation of Twitter US Airline Sentiment Tweets from English to Romanian is highlighted as the first and most important step

[11] `https://sklearn-genetic-opt.readthedocs.io/en/stable/api/gasearchcv.html`
[12] `https://deap.readthedocs.io/en/master/`

Figure 3.1: Architecture of the Sentiment Analysis system

required to run SA in different languages using a single uni-lingual dataset as the source of information.

The preprocessing step consists of various procedures and are grouped in two different abstract pipelines. The one on the left will generate texts which are fit for the TF-IDF variants, Doc2vec and Word2Vec techniques. In the one on the right only steps 3 and 5 from our preprocessing module are applied, plus a sentence level tokenization in order to generate texts as expected by the pretrained BERT encoder.

The TF-IDF variants and Doc2Vec are grouped together to highlight that the output of all these methods are in the same form. To be more specific, a preprocessed text is transformed into a vector of length N while Word2Vec will transform a preprocessed text into a NxM matrix where the number of rows will be equal to the number of words within the text, and the number of columns will be equal to the word embedding size. BERT contextual encoding will represent texts using multiple vectors with the help of Multilingual Tokenizer.

In model training and tuning, we can see that for all the selected ML approaches, with the exception of M-BERT, evolutionary hyperparameter optimization is used to identify the best set of parameters. We selected genetic algorithms over grid search to avoid the pitfall of finding local minima and because GAs are able to explore the search space of parameters with continuous values. Due to the high training times of BERT, we opted for a classic training process using the recommended parameters.

Bernoulli NB, Linear SVM, LR, RF and DNN are grouped together to highlight the many-to-many relation of this group with the TF-IDF variants and Doc2Vec features. This means that any feature from this group can be used by any model mentioned previously. LSTM and CNN are grouped together in order to highlight that both of them use the Word2Vec features, while the BERT classifier uses the specific encodings generated by its own Multilingual Tokenizer.

At the bottom of the diagram, the main goal of our work is highlighted, namely the classification of input texts in 3 categories: negative, neutral, or positive.

Next, we present our experiments and discuss the results. The classifiers will be evaluated strictly on the test sets using 3 metrics often used in the literature: Macro F1, Weighted F1, and Accuracy. In addition to this, we will present and analyze the execution speeds of each ML model. With our findings at hand, other researchers can more easily select the model that suits their needs based on the expected predictive performance in relation to hardware usage.

All the experiments, with the exception of fine tuning Multilingual BERT, were conducted on a high performance computer with the following specifications: 2 $\times$ Intel Xeon Gold 6230 CPUs, 128 GB of DDR4 RAM, and 8 $\times$ NVIDIA Tesla V100 GPUs with 32GB of VRAM each. For BERT we used a development environment equipped with Tensor Processing Units (TPUs) provided to us by Google. We did not run BERT on our powerful computer because after some investigations we discovered that we can actually get faster execution times on the TPUs. The source code was implemented in Python 3.9.

We applied the processing pipeline described in Figure 3.1 on the original and translated Twitter US Airline Sentiment Tweets dataset. We used the same exact methodology for both languages for consistency but most importantly to isolate the impact of machine translation from other variables.

Table 3.2 shows the learning performances of the classifiers and for the algorithms that were paired with the TF-IDF encoding the results with dimensionality reduction are also provided.

Encoding	Classifier	English original dataset			Romanian translated dataset		
		Acc.	Weighted F1	Macro F1	Acc.	Weighted F1	Macro F1
TFIDF	Bernoulli NB	77.37	77.25	70.7	78.2	78.2	71.91
	Linear SVM	77.41	76.67	69.77	78.36	77.47	70.54
	RF	77.45	75.9	68.47	77.81	76.45	69.04
	LR	66.7	55.21	39.2	65.2	54.71	38.17
	DNN	78.18	77.15	70.14	77.2	76.23	69.19
TFIDF+PCA	Bernoulli NB	68.01	62.82	50.74	67.78	62.49	50.4
	Linear SVM	75.89	75.94	69.52	76.2	74.71	66.94
	RF	75.78	74.79	67.28	75.73	73.47	65.05
	LR	67.52	58.95	45.39	63.66	57.16	42.17
	DNN	76.34	75.34	68.43	76.58	75.75	68.59
TFIDF+NMF	Bernoulli NB	72.44	71.7	63.43	72.93	72.22	64.43
	Linear SVM	74.14	74.13	67.1	73.16	69.88	60.5
	RF	74.77	73.27	65.89	74.99	73.88	66.05
	LR	62.55	48.01	25.62	65.41	55.17	39
	DNN	74.67	74.28	66.66	75.02	73.07	64.7
TFIDF+LSA	Bernoulli NB	68.26	63.97	52.23	67.01	60.77	47.43
	Linear SVM	76.3	75.05	67.31	76.9	75.65	68.18
	RF	76.12	74.29	66.33	76	74.3	66.23
	LR	67.81	68.72	61.72	64.1	57.87	43.15
	DNN	76.67	75.89	69.06	76.36	75.07	67.67
Word2Vec	CNN	78.21	76.66	70.33	77.69	76	68.67
	LSTM	77.5	76.35	69.4	78.17	77.98	71.39
Doc2Vec	Bernoulli NB	62.44	48.01	25.62	62.42	47.98	25.62
	Linear SVM	63.05	48.01	25.62	62.67	47.97	25.62
	RF	62.75	48	25.22	62.52	47.97	25.62
	LR	62.44	47.9	25.02	62.42	47.79	25.62
	DNN	62.9	48.01	25.62	62.44	47.98	25.62
Multilingual BERT Tokenizer	Multilingual BERT	83.02	82.57	77.48	80.99	80.5	74.81

Table 3.2: Classifier predictive performance

One of the most important insights from Table 3.2 is that the classification performance between languages is surprisingly consistent. For example, Bernoulli NB with plain TF-IDF shows a slight improvement in accuracy from 77.37% in English to 78.2% in Romanian. Linear SVM and DNN also show only a minor variation between languages, suggesting that the translation learning approach is viable for Romanian. The variation among all combinations of models and encoding schemes is around $\pm 2\%$ in terms of accuracy, and this insignificant difference is also maintained for Weighted and Macro F1.

Multilingual BERT sets the state of the art accuracy at 83% on the English dataset and 81% on the Romanian dataset. In the case of classic ML, Bernoulli NB and Linear SVM had the best results across all 3 evaluation metrics, reaching an accuracy of approximately 78% for both languages. RF has a comparable accuracy but a greater decrease in performance when we consider the Weighted and Macro F1 scores. The LR was considerably worse than the rest in this group, denoting its inability to capture the necessary information for correct classification.

DNN, CNN, and LSTM performed similarly to Bernoulli NB and Linear SVM with accuracies of around 78% in English and 77–78% in Romanian. DNN with TF-IDF and CNN with Word2Vec have slightly better results on the English set while LSTM with Word2Vec is a little better on the Romanian one, but the differences are negligible.

Another important finding we want to highlight here is the impact of dimensionality reduction on TF-IDF vectors: PCA, NMF, and LSA decreased the classification performance of the models when compared to the same models that used the original vectors. For example, Linear SVM's accuracy in Romanian dropped from 78.36% with TF-IDF to 76.2% with TF-IDF+PCA, 73.16% with TF-IDF+NMF, and 76.9% with TF-IDF+LSA. A similar trend can be seen in English. This suggests that although DR algorithms may improve execution speed, they also negatively impact accuracy rates due to information loss. However, we note that the impact is not so drastic, which means that a trade-off like this could be considered acceptable depending on the context of the application.

By far the worst performances are those of the models using Doc2Vec as the encoding mechanism. Compared to TF-IDF which is based on word frequency and Word2Vec which builds a dedicated vector for each individual word depending on its context, Doc2Vec attempts to generate a single dense vector to represent each instance from the dataset. This technique might work well with longer texts but it's problematic in our case because the shorter the texts are, the greater the risk of injecting errors in the generated vectors becomes. This most likely explains Doc2Vec's poor performance. Regardless, no model using Doc2Vec achieved predictive results acceptable for real-world applications.

With this analysis we note that although the translation from English to Romanian comes with certain variations, the performances of all selected classifiers remained stable between languages. This confirms that an automated translation approach can be used to create resources for SA in Romanian.

In Table 3.3 we list the execution times for hyperparameter optimization, training of the final models, and testing them. With the help of evolutionary optimization we managed to increase the weighted F1-measure of the models by 1–3%, with slightly greater gains for accuracy.

As a first observation, we want to point out the higher optimization and training times that were needed on the Romanian dataset. This is not surprising because the Romanian language tends to be more verbose than English, which means that a greater number of words are used to express the same ideas. For this reason, the Romanian models end up with a larger vocabulary than the models trained on the English dataset. However, this increase in learning times for the Romanian language is not big enough to be considered a real obstacle in practice.

Another aspect that can be observed is related to the great variations in the optimization and training speeds across classifiers and encodings. Obviously, training classic ML models is faster than DL models because the latter are more complex by nature. Even so, the times to train the final models are pretty insignificant around the board with the exception of Multilingual BERT which took 7 minutes for each language. Second worse but considerably faster is LSTM, with 17 seconds for English and 21 seconds for Romanian. The fastest was Bernoulli NB, needing less than 1 second for all language and encoding combinations.

The problem arises when searching for optimal model parameters, as this is very time-consuming: model complexity leads to more parameters, a larger search space, and longer training times; factors that together cause optimization times to grow exponentially. For Bernoulli NB without dimensionality reduction this took about 27 minutes in Romanian and 22 minutes in English. Linear SVM had similar times between languages, slightly over 16 minutes. Searching for the best parameters and network capacity of the DNN took about 3 hours and 45 minutes on the Romanian data and 3 hours and 11 minutes on the English data. For LSTM this process took more than 17 hours for Romanian and

Encoding	Classifier	English original dataset			Romanian translated dataset		
		Opt. (s)	Train (s)	Test (s)	Opt. (s)	Train (s)	Test (s)
TFIDF	Bernoulli NB	1337	0.285	0.128	1645	0.368	0.147
	Linear SVM	920	0.36	0.02	1048	0.238	0.022
	RF	89603	8.502	0.024	3588	7.855	0.026
	LR	5735	1.59	0.085	6158	0.568	0.122
	DNN	11513	2.176	0.274	13551	2.23	0.44
TFIDF+PCA	Bernoulli NB	395	0.051	0.03	387	0.064	0.031
	Linear SVM	2450	2.089	0.016	2038	2.418	0.01
	RF	2113	14.044	0.02	2245	4.792	0.022
	LR	795	1.587	0.134	375	0.295	0.05
	DNN	10195	1.561	0.19	3558	1.015	0.166
TFIDF+NMF	Bernoulli NB	384	0.058	0.044	362	0.051	0.017
	Linear SVM	482	0.276	0.015	412	0.149	0.013
	RF	226	1.885	0.014	5125	16.644	0.019
	LR	567	0.479	0.077	449	0.404	0.03
	DNN	17842	5.109	0.207	9236	5.28	0.176
TFIDF+LSA	Bernoulli NB	389	0.064	0.016	385	0.062	0.032
	Linear SVM	8417	6.53	0.003	2724	2.353	0.009
	RF	2245	0.743	0.028	204	0.693	0.022
	LR	943	1.546	0.14	659	0.905	0.154
	DNN	10195	1.782	0.214	6585	2.567	0.261
Word2Vec	CNN	9143	1.46	0.281	16127	4.209	0.25
	LSTM	17172	16.865	1.32	62364	20.926	1.292
Doc2Vec	Bernoulli NB	271	0.028	0.01	274	0.021	0.015
	Linear SVM	654	0.19	0.013	580	0.152	0.014
	RF	484	1.595	0.015	428	0.936	0.01
	LR	712	1.171	0.14	263	0.523	0.111
	DNN	17842	0.912	0.144	11529	4.594	0.279
Multilingual BERT Tokenizer	Multilingual BERT	N/A	416.13	16.64	N/A	444.02	16.73

Table 3.3: Classifier hyperparameter optimization, training, and evaluation times

almost 5 hours for English, despite the vector embedding size being only 200. Among the classical ML models, RF and LR were the slowest.

Testing times are relatively low for all models and encodings pairs which shows that once trained the classifiers are able to quickly deliver a large number of predictions regardless of language. Multilingual BERT, despite its long training, achieved good evaluation times of 16.64 and 16.73 seconds for the English and Romanian datasets. The second slowest was LSTM but it only needed around 1.3 seconds to finish. The other models (including CNN) have even better results, with test times of under 0.5 seconds in both languages.

Overall, these execution times provide helpful insights about how demanding our selected ML algorithms are. A complex model like BERT has higher predictive capabilities but it's slower to run, while simpler models like Bernoulli NB or Linear SVM offer a reasonable compromise between speed and accuracy which makes them ideal in cases where hardware resources are limited.

Chapter 4

Topic Classification for Romanian Social Media Texts

For our work we require a large dataset with social media specific texts. The number of available topics within the dataset should cover the general but noteworthy topics and the topic label of each instance should be correctly assigned, preferably manually annotated. Unfortunately, we couldn't find a single publicly available dataset that could satisfy all of our research necessities and we don't have the available resources to collect and label a new one from scratch. Thus, we decided to translate an open source English dataset to Romanian with automated translation and use it as "surrogate" in our experiments [48].

We selected the News Category Dataset[1] which contains 202,372 news headlines collected between 2012 up to 2018 from HuffPost[2]. This site offers news, satire, blogs, original content, and covers a variety of topics. Each record of the dataset contains the following attributes: *category* (41 categories), *headline*, *short_description*, *authors*, *date* (of the publication), and *link* (URL link of the article).

There are a number of reasons why we selected this dataset as the benchmark for our experiments: (i) It contains short texts similar to those found on social media platforms (ii) The topics are fairly general and the number of topics is large enough (iii) The category of each article was manually labeled (iv) The dataset is large enough to effectively train the ML models (v) It was relatively recently collected.

For our classification problem we'll focus only on the headline and short description attributes of the dataset, ignoring the authors and date of publication. Therefore, we merged the headline and the short description attributes and created a novel attribute called *text_merged*. The vast majority of merged texts contain between 94 and 254 characters, with the mean being ≈174 and the standard deviation almost 80 characters. This proves that the generated texts have the characteristics of short texts similar to those present in social media platforms (i.e. a Twitter tweet is limited to 280 characters, a TikTok comment is limited to 150 characters).

Next, we did an initial investigation of the data and encountered some problems with the distribution and granularity of the original 41 class labels, as depicted in Figure 4.1. The top-3 most popular classes are: "POLITICS" with ≈ 16% of the records, "WELLNESS" with ≈ 9% of the records, and "ENTERTAINMENT" with ≈ 8% of the records. The least most popular 4 classes are: "COLLEGE", "LATINO VOICES", "CULTURE & ARTS", and "EDUCATION" each making up only 0.5% of the records, meaning that

[1] `https://www.kaggle.com/datasets/rmisra/news-category-dataset`

[2] `https://www.huffpost.com/`

there is a significant class imbalance in the data.

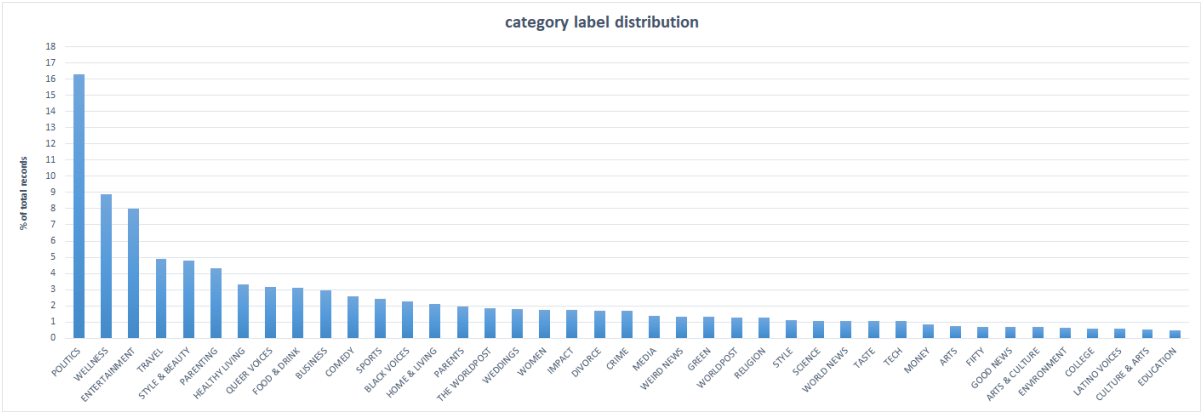

Figure 4.1: Original topic category distribution, 41 classes

We also noticed that there are 2 more issues with the existing topics: a subset of them are overlapping and others are way too granular. For example the categories "SCIENCE" and "TECH" are too specific but can be naturally grouped together in a common class like "SCIENCE & TECH", while other classes have different labels but denote the same thing, for example "ARTS & CULTURE" and "CULTURE & ARTS".

So, in order to improve the quality of the data, we decided to cluster together the overly granular and synonymous categories. Therefore, we transformed the next classes as follows: "HEALTHY LIVING" was relabeled as the existing "WELLNESS" class; "PARENTS" was relabeled as the existing "PARENTING" class; "STYLE" was relabeled as the existing "STYLE & BEAUTY" class; "GREEN" was relabeled as the existing "ENVIRONMENT" class; "TASTE" was relabeled as the existing "FOOD & DRINK" class; "COLLEGE" was relabeled as the existing "EDUCATION" class; "THE WORLDPOST" and "WORDPOST" were relabeled as the existing "WORLD NEWS" class; "ARTS" and "CULTURE & ARTS" were relabeled as the existing "ARTS & CULTURE" class; "BUSINESS" and "MONEY" were relabeled as a new class named "BUSINESS & FINANCES"; "SCIENCE" and "TECH" were relabeled as a new class named "SCIENCE & TECH"; "QUEER VOICES", "BLACK VOICES", and "LATINO VOICES" were relabeled as a new class named "GROUPS VOICES"; "FIFTY" and "GOOD NEWS" were relabeled as a new class named "MISCELLANEOUS".

At the end of this process the adjusted dataset contains 26 topics that are truly distinct and no class has less than 1% of record labels, meaning that the least popular class has more than 2,000 records. This should increase the performance of the models that will be trained later but at the same time it ensures consistency and coherence in our topic classification task. This new class feature was named *category_merged* and its full distribution is shown in Figure 4.2.

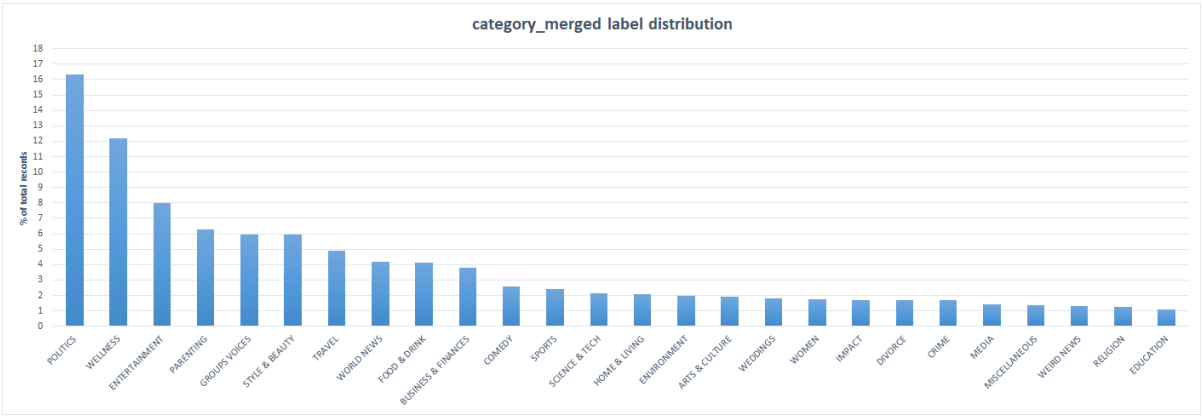

Figure 4.2: Merged topic category distribution, 26 classes

For our experiments, the dataset was split into training and test sets using a standard 75–25% split: ≈151,500 instances for training and ≈50,500 for testing, with similar class distributions in both. Moreover, the English and Romanian train and test data are identical in the sense that they contain the same set of instances.

In order to remove the natural noise which is existent in our dataset we used the following 5 preprocessing steps, applied in this specific order:

1. Extra white space removal (language-independent).
2. Word lemmatization and tokenization (language-dependent).
3. Stop-word removal (language-dependent).
4. Lower case capitalization (language-independent).
5. Punctuation mark removal (language-independent).

If the predictions are made on real life social media texts, then a number of additional preprocessing steps are required in order to bring them to a closer format similar to that existent in the training set. Because the texts of the News Category dataset do not manifest these characteristics we'll not go into any further details, but more information about these processing steps can be found in Chapter 3.

For this study, we selected two of the most popular approaches as suggested by the NLP literature [25, 39]: TF-IDF and Word2Vec.

We won't include Doc2Vec as a feature extraction method for two reasons. First, the classifiers that used this type of embedding had by far the worst predictive performances in the SA experiments, as shown in Chapter 3. Secondly, the topic classification dataset is considerably larger than the one we used for SA, containing around 202,000 instances compared to 15,000. This is equivalent to a 13.5 fold increase in the volume of data which would lead to much higher execution times for hyperparameter optimization, model training, and testing.

We trained one TF-IDF vectorizer on the original English training set and another one on the Romanian translation. TFIDF was applied on the preprocessed token lists and the vocabulary was set to contain the tokens which appear at least 5 times. This was done in order to remove a large number of tokens which are very rarely used or tokens which may have been erroneously generated in the preprocessing step. Next, the newly created TF-IDF models were executed on the test sets, after which we noticed that the English

vocabulary contains around 25,000 tokens whereas the Romanian vocabulary contains 27,500 tokens but we expected this slight difference because Romanian is more verbose than English.

Due to the sparse nature of TF-IDF, the large number of training instances, and the vocabulary sizes, the trained vectors are stored and used in the Compressed Sparse Row (CSR) format. CSR is advantageous for handling sparse matrices because it efficiently compresses the storage of non-zero elements, significantly reducing memory usage and improving performance when performing matrix/vector multiplications [24].

To learn the Word2Vec embeddings we used Gensim library [64] and, as in the case of TF-IDF, we had to create two dedicated models, one for each language. We selected the Continuous Bag-Of-Word (CBOW) architecture as it works better for short texts and set the vocabulary to include only tokens that appear at least 5 times. The token embedding size was set to 300 to balance contextual understanding and runtime efficiency. The Word2Vec models were trained with the following parameters: learning rate (α) of 0.025, a window size of 5, over 5 epochs.

We decided against reducing the TF-IDF vector dimensions because we experimented with this process for SA and found that algorithms like LSA, NMF or PCA are indeed able to improve execution speeds but they also decrease the predictive capability of the models. Given the importance of maintaining a high accuracy in our topic classification task this performance trade-off isn't justifiable, especially since we have a much larger number of classes here.

For our experiments we selected the following learning methods:

- Classic ML:
  - Bernoulli Naive Bayes (Bernoulli NB)
  - Support Vector Machine with a linear kernel (Linear SVM)
  - Random Forest (RF)
- Deep Learning:
  - Long Short-Term Memory (LSTM)
  - Convolutional Neural Network (CNN)
- Advanced Language Model:
  - Multilingual BERT

We decided not to include LR and DNN here because these models were used in the SA study and didn't achieve any outstanding results. LSTM and CNN will be applied on the Word2Vec features due to their ability to process sequential data, while the classic learning algorithms will be applied on TFIDF. In order to run LSTM and CNN on sequenced data, we had to introduce an extra embedding layer right after the input layer. This embedding layer maps each token from an instance to its corresponding Word2Vec representation, being equipped with the word embeddings generated during feature extraction.

For BERT, we used the base multilingual uncased variant available on the Hugging Face transformers[3]. On top of M-BERT we added a hidden dense layer with 128 nodes and ReLU activation function, followed by a standard classification layer with 26 nodes which produces the topic class. Adam was the selected optimizer, with a learning rate of 2×10^{-5} and $\epsilon = 10^{-8}$. The loss function was set to Categorical CrossEntropy.

Our evolutionary optimization algorithm for TC was designed as follows. Having a number N of parameters to optimize, an individual/chromosome is denoted as a vector

[3] `https://huggingface.co/docs/transformers/en/model_doc/bert`

$(p_1, p_2, \ldots, p_i, \ldots, p_N)$. Within this vector, each p_i represents the value selected for the corresponding hyperparameter N_i. The population consists of 10 individuals, each representing a potential parameter combination solution. This population is evolved over 20 generations with a crossover probability of 80% in order to combine the characteristics of the individuals and a mutation probability of 10% to introduce variation in the population. Individuals are selected for the next generation using a standard elitist tournament of size 3. Internally, each individual is evaluated using the accuracy metric as the fitness function, computed with 3-fold cross-validation. In general, convergence can be seen after 10-15 generations, thus evolving the populations over 20 generations is enough to guarantee a good parameter selection.

In the case of classic ML all the parameters described in the official Sklearn documentation were optimized. In the case of the LSTM and CNN, we considered among the parameters the following: the network capacity (the number and size of hidden layers), the activation function, the regularization function, drop-out rate. For M-BERT, as learning just one model is very time consuming, we omitted to perform the hyperparameter optimization procedure. Instead of cross-validation, we reserved 10% of the training set for validation and we let the model to optimize the loss function for several epochs. We noticed that for both languages the maximum accuracy on the validation set is reached at 5 epochs, so we stopped the training process at this point.

Next, we present the high level architecture design of our Topic Classification system. Figure 4.3 summarizes all the procedures and steps involved.

At the top of the diagram, the automatic translation of the News Category dataset from English to Romanian is shown as the first key step for enabling TC in different languages using a single uni-lingual dataset as the source of information.

The preprocessing step consists of various procedures and are grouped in two different abstract pipelines. The one on the left will generate texts which are fit for the TF-IDF and Word2Vec techniques. In the one on the right only a sentence level tokenization is applied in order to generate texts as expected by the pretrained BERT encoder.

The feature extraction step highlights the transformation of the preprocessed text into numeric features which can be later used to train the ML models. With TF-IDF, a preprocessed text is transformed into a vector of length N while Word2Vec will convert the same preprocessed text into a NxM matrix, where the number of rows will be equal to the number of words within the text and the number of columns will be equal to the word embedding size. BERT contextual encoding will represent texts using multiple vectors with the help of Multilingual Tokenizer.

In model training and tuning, we can see that for all the selected ML models, except BERT, the evolutionary hyperparameter optimization methodology is used to find the best set of parameters. We selected genetic algorithms over grid search to avoid getting stuck in local minima and because GAs are able to explore the search space of parameters with continuous values. Due to the high training times of BERT, we opted for a classic training process using the recommended parameters.

Bernoulli NB, Linear SVM, and RF are grouped together to highlight the many-to-one relation of this group with the TF-IDF features. LSTM and CNN are grouped together in order to highlight that both of them use the Word2Vec features, while the BERT classifier uses the specific encodings generated by its own Multilingual Tokenizer.

At the bottom of the diagram, the primary goal of our study is highlighted, namely the classification of input texts into 26 discussion topics. The complete list of all topics, ordered from most common to least common is: "politics", "wellness", "entertainment",

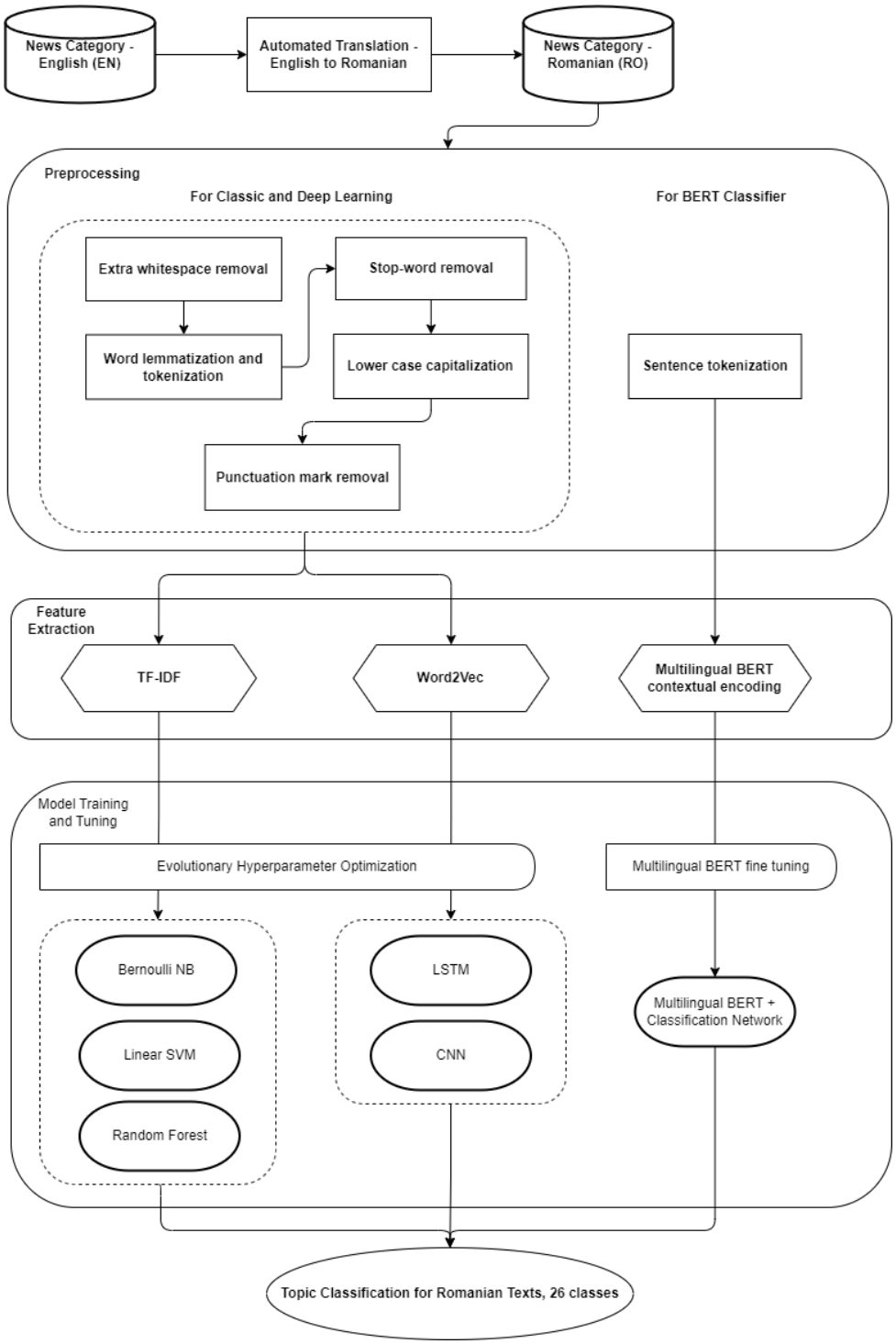

Figure 4.3: Architecture of the Topic Classification system

"parenting", "groups voices", "style & beauty", "travel", "world news", "food & drinks", "business & finances", "comedy", "sports", "science & tech", "home & living", "environment", "arts & culture", "weddings", "women", "impact", "divorce", "crime", "media", "miscellaneous", "weird news", "religion", "education".

Next, we showcase the experiments and discuss their implications. The models will be evaluated strictly on the test sets using the standard accuracy (Top-1), as well as Top-2 and Top-3. The Top-K accuracy measure is very useful given the large count of topic classes and the potential for topic overlap within texts [26, 52]. Additionally, we'll

present and analyze the execution speeds of each ML model. All the experiments were conducted on the same hardware as described in Chapter 3.

We applied the processing pipeline described in Figure 4.3 on the News Category datasets, using the same exact methodology for both languages for consistency but most importantly to isolate the impact of automated translation from other variables.

One of the most important insights from Table 4.1 is that the predictive performance of Multilingual BERT, Bernoulli NB, and Linear SVM between the two languages is surprisingly consistent, the variations being only around 2–3% for all considered Top-K values. CNN showed a more noticeable drop on the Romanian dataset, with decreases of around 4% for Top-1 and Top-2, and 5% for Top-3, which denotes a moderate sensitivity to language translation. On the other hand, LSTM had the biggest drop, with decreases of 14% in Top-1, 15% in Top-2, and 13% in Top-3, which shows that its effectiveness was seriously hampered by the translation process. RF experienced a similar drop of around 13–14% across all metrics, indicating its limited adaptability on the translated data.

Encoding	Classifier	English original dataset			Romanian translated dataset		
		Top-1	Top-2	Top-3	Top-1	Top-2	Top-3
TFIDF	Bernoulli NB	64.17	78.93	85.27	62.80	77.70	84.11
	Linear SVM	67.97	81.75	87.10	66.73	80.05	85.30
	RF	30.0	40.74	50.21	16.56	28.89	37.0
Word2Vec	CNN	66.15	80.07	85.24	61.66	74.05	79.28
	LSTM	67.64	80.41	85.55	53.50	65.59	72.39
Multilingual BERT Tokenizer	Multilingual BERT	74.85	87.29	91.73	72.63	85.56	90.25

Table 4.1: Classifier predictive performance

Multilingual BERT has state of the art results with an accuracy of 74.85% Top-1, 87.29% Top-2, and 91.73% Top-3 on the English dataset, and 72.63% Top-1, 85.56% Top-2, and 90.25% Top-3 on the Romanian dataset.

In the case of classic ML, Linear SVM had the best results and is the second-best overall after Multilingual BERT. It maintained a high predictive consistency between English and Romanian by achieving approximately 68%, 82%, and 87% on the English dataset and 67%, 80%, and 85% on the Romanian one. Bernoulli NB is the second-best model in this category, and third place overall, with Top-K scores between 2–3% lower than Linear SVM on the English dataset. On the Romanian dataset, the same 2–3% difference holds true for Top-2 and Top-3, but for Top-1 a greater decrease of approximately 4% is seen, meaning that Linear SVM is preferable when the main topic matters most for the analysis.

RF had by far the worse performance with only 30%, 40%, and 50% accuracy in English, and 17%, 29%, and 37% in Romanian. To put this into perspective, this means that even the Top-3 accuracy is considerably lower than the Top-1 accuracy of Bernoulli NB which is very surprising because RFs have been employed with success in NLP for many classification tasks. Even in our SA experiments, the RF worked well for both languages, being on par with the best performing classic learners, but the results obtained here clearly show that something went wrong. The biggest difference compared to SA is that in this case we have a much larger number of target classes (26 compared to 3). RF splits data using decision trees to maximize information gain, but having many classes can

lower split "purity" as it's harder to find enough linguistic features to clearly distinguish between abstract topics of discussion.

CNN initially outperformed Bernoulli NB in English with accuracies of 66% Top-1, 80% Top-2, and 85% Top-3 but experienced a more significant drop in Romanian, performing 2–4% worse than Bernoulli NB, reducing its score to 62%, 74%, and 79% respectively. LSTM had results similar to CNN on the English texts, even surpassing CNN in Top-1 prediction by almost 2%, but as stated previously it had the biggest drop on the Romanian texts, achieving only 54% Top-1, 66% Top-2, and 72% Top-3.

With this analysis we can note that although the translation from English to Romanian comes with certain variations and changes, the performances of most of the classifiers subjected to this kind of experiment remain stable. This confirms that an automated translation approach can be used to create the necessary resources for TC in Romanian.

In Table 4.2 we list the execution times for hyperparameter optimization, training of the final models, and testing them. With the help of evolutionary optimization we managed to increase the Top-1 accuracy of the models by 2–5%.

Encoding	Classifier	English original dataset			Romanian translated dataset		
		Opt. (s)	Train (s)	Test (s)	Opt. (s)	Train (s)	Test (s)
TFIDF	Bernoulli NB	443	0.567	0.035	389	0.59	0.04
	Linear SVM	8005	25.798	0.034	11803	45.91	0.042
	RF	2992	14.593	0.1	845	0.6	0.183
Word2Vec	CNN	37317	46.493	1.58	36797	56.98	1.65
	LSTM	286062	130.19	9.5	63605	119.1	6.16
Multilingual BERT Tokenizer	Multilingual BERT	N/A	7420	157	N/A	7498	157

Table 4.2: Classifier hyperparameter optimization, training, and evaluation times

First of all, we can note that the times for hyperparameter optimization vary a lot depending on the language and classifier. Compared to the SA experiments where most models had slower training and optimization times for Romanian, here we can see that things are more balanced. The reason why 4 out of the 5 models have better optimization times on the Romanian texts is most likely due to a faster convergence of the algorithms in the hyperparameter optimization stage. This makes sense if we check the training times because in this case only 2 models, namely LSTM and RF, were faster in Romanian.

The optimization of Linear SVM was substantially slower than the rest of classical ML approaches: on the English dataset it took ≈2.2 hours and on the Romanian dataset it took ≈3.3 hours. These extended periods of time can be attributed to the high number of parameters in the SVM model family, which means that more evolutionary iterations were needed to reach optimal performances. Another factor that could have increased the times even more for Romanian is the increased count of tokens in the vocabulary.

RF exhibited a much shorter optimization time for Romanian, taking only ≈14 minutes compared to ≈50 minutes for English. These low times paired with the poor predictive performance in both languages suggests that the hyperparameter optimization process wasn't able to find a good set of parameters, missing both global and local optima. In contrast, Bernoulli NB was by far the fastest to optimize: ≈7.4 minutes for English and ≈6.5 minutes for Romanian.

Similar to hyperparameter optimization, the training times differ from model to model with Bernoulli NB being extremely fast (0.567 seconds for English and 0.59 seconds for

Romanian) due to its simple probabilistic implementation. Linear SVM is considerably more computationally demanding but it managed to finish training in a reasonable time frame, needing 25.79 seconds for English and 45.91 seconds for Romanian. RF again has a mixed performance, with training times of 14.593 seconds in English and only 0.7 seconds in Romanian.

As expected, the DL models had much higher run times when compared with the classic learners, both in the case of their optimization and their training. CNN required ≈10.4 hours for optimization on the English dataset and ≈10.2 hours on the Romanian dataset, with training times of 46.49 seconds and 56.98 seconds, respectively. LSTM was even slower, its optimization taking an incredible ≈79.5 hours for English and a more realistic ≈17.7 hours for Romanian. The significantly shorter time on the Romanian dataset paired with a big drop in predictive performance for this language suggests that the hyperparameter optimization process may have found a local optimum rather than a global one. The training times of 130.19 seconds for English and 119.1 for Romanian on the other hand can be considered acceptable.

Although higher than the other models, Multilingual BERT's execution times are very similar between the two languages, training taking ≈2.1 hours and testing 157 seconds for each language. This consistency aligns with BERT's architecture which is designed to handle multilingual data but at a higher computational cost. The absence of hyperparameter tuning is offset by BERT's pretraining on large volumes of diverse data, allowing us to achieve superior predictive results with standard parameters.

The test times are relatively low for all models and encodings pairs, considering the generous size of the test set which consists of around 51,000 instances. This indicates that once trained, the classifiers can quickly deliver a significant number of predictions regardless of language. Multilingual BERT, despite its slow training, has much more acceptable evaluation times. The second slowest was LSTM, with 9.5 seconds for English and 6.16 seconds for Romanian. CNN achieved better times of under 1.7 seconds for both datasets while the classic ML models were the fastest by a considerable margin, all of them finishing in less than 0.2 seconds.

Overall, these execution times provide helpful insights about how demanding our selected algorithms are. A complex model like BERT has higher predictive capabilities but it's slower to run while simpler models like Linear SVM or Bernoulli NB offer a reasonable compromise between speed and accuracy, which makes them ideal in environments that lack sufficient computational resources.

Chapter 5

BERTweetRO: Language Models for Romanian Social Media Texts

In this chapter we present our efforts in pre-training from scratch 8 BERTweetRO models, based on RoBERTa architecture, with the help of a Romanian corpus containing public tweets. To evaluate our models we fine-tune them for Sentiment Analysis (with 3 polarity classes) and Topic Classification (with 26 classes), and compare them against Multilingual BERT plus a number of other popular classic and deep learning models [47].

Twitter Stream[1], collected by Archive Team, is a valuable public corpus that offers a huge volume of texts that were scrapped from Twitter and stored in JSON format. It covers all the years starting from 2012 until the middle of 2021, split into 2,900 files that amount to ≈6.8 TB of data. The exact number of tweets in this dataset is not specified, but by considering the long time frame that it covers plus the size of the documents we can say, with a high degree of certainty, that Twitter Stream should satisfy a large range of objectives. Researchers, private or public institutions could use this data to analyze trending topics, public sentiments, cultural or social events, and more in real time or in retrospect to answer questions about the dynamics of modern societies.

As opposed to other resources this one is not limited to include only tweets in internationally popular languages because, in the web-scraping process, the majority of public posts were collected, regardless of their language. Thus, we'll use the Twitter Stream to pretrain our custom BERT models as it captures the evolution of Romanian language, and the way it's used, in a microblogging context. Given the size of the entire tweet archive, our limited hardware dictates a need for data selection in order to train multiple versions of RoBERTa models in a reasonable timeframe. With respect to this, we decided to only use a subset of Twitter Stream that encompasses approximately 800 GB of data spanning over the course of one year: July 2020 through June 2021.

Another factor that led to this decision relates to the fine-tuning tasks that are going to be made on the newly created RoBERTa models, namely SA and TC, and by acknowledging this constraint we aim to establish a Proof of Concept (POC) demonstrating the feasibility of training BERT based models on Romanian social media texts using a relatively small dataset. This will likely result in lower performance compared to using the whole archive, but our ultimate goal is to show that it's possible for researchers to create decent models in cases where there are strong hardware or time limitations. In future iterations, if more computational resources become available, we hope to include the remaining data in the pretraining pipeline to develop even stronger models.

[1] `https://archive.org/details/twitterstream`

As a first step, we downloaded the data from the target period mentioned earlier after which we performed a manual inspection to familiarize with the structure and nature of it. The JSON documents contain 2 different types of instances, one denoting the removal of content from the platform and includes the ID of the deletion plus some other metadata but without useful textual information. The other type, referred to as "post", contains a lot of information but of interest to our study are the "text" field, which represents the tweet message, and the "lang" field, which indicates the language of the message.

Next, we selected and examined in more detail 200 random posts from a 2 month period and discovered a major problem with "lang": a number of tweets were labeled as Romanian when in reality they weren't. Many were simply misclassified, in some extreme cases as a very different language like Malay, and others were pure "noise" posts that only contained a mix of Twitter mentions, hashtags, URLs, and emojis. This highlights the problems that can appear when dealing with online user generated content where the informal tone of communication, errors in grammar, and other irregularities are degrading the accuracy of automated language identification tools.

Following this initial investigation we decided to use Python[2] together with langid[3] to correctly identify the language of the posts. We selected langid because it's been trained on a wide number of languages (currently supporting 97 in total), which makes it a good choice for our multilingual dataset, and it offers very fast processing times paired with state of the art results. Another advantage of lanid is that it offers a "confidence level" score for each prediction that acts as a measure of reliability. We ran langid on the same subset of 200 tweets using a high threshold approach in which we consider the texts to be Romanian only if the confidence level exceeds 95% to avoid incorporating false positives. We made a second review of the language classification and saw that most of the texts were labeled correctly this time around but some outliers still persisted.

The performance on raw texts was satisfactory, but to improve results we implemented a preprocessing pipeline that includes the automatic identification and removal of Twitter mentions, Twitter hashtags, URL links, and emoticons. With this mechanism in place we want to deliver cleaner and more standardized texts to langid in the hope of improving the accuracy. We ran langid once again but this time on the cleaned data and performed another round of investigations. The results were clearly better which means that the proportion of tweets correctly labeled as Romanian has increased, thus validating our custom language identification framework.

Table 5.1 shows that over a period of 12 months we identified and extracted around 51,000 tweets posted in Romanian which means that we have ≈4,250 tweets for each month on average. While this dataset may seem small at first glance, we argue that it's sufficient to provide a relevant snapshot of the activity of Romanian speakers on Twitter. Because of preprocessing and language identification the total execution time for this extraction process was very high, totaling to over 72 hours.

This relatively modest number of extracted tweets also aligns with the low number of Romanian Twitter users. According to Statista[4], the number of Twitter users in Romania was around 600,000 during the time period targeted by us. It is also important to note that not all users make their posts public and additionally some accounts might have privacy settings in place. These aspects, along with certain geographical or other restrictions, mean that part of the user generated content may have been skipped during the scraping

[2] `https://www.python.org/`

[3] `https://pypi.org/project/langid/`

[4] `https://www.statista.com/forecasts/1143811/twitter-users-in-romania`

Year-Month	Number of texts labeled as Romanian in Twitter Stream	Number of texts labeled as Romanian with our approach	Percent Romanian	Execution Time (Hours)
2020-07	48415	4256	8.8	6.4
2020-08	56292	5100	9.06	7.7
2020-09	59346	4729	7.97	7.3
2020-10	57778	4788	8.27	7.5
2020-11	48867	4406	9.02	5.5
2020-12	52896	4935	9.33	6.1
2021-01	22621	1771	7.83	2.5
2021-02	56163	4621	8.23	6.21
2021-03	57993	5210	8.98	6.7
2021-04	24149	2095	8.68	2.7
2021-05	58576	4702	8.03	7.2
2021-06	52475	4330	8.25	6.3

Table 5.1: Comparison of Romanian labeled tweets

of Twitter Stream.

For Romanian, several studies have tackled the task of creating language models by leveraging the transformer architecture together with large scale datasets to increase the level of automated language understanding and generation. Here we can mention the works of Dumitrescu et al. [21] who introduced the first purely Romanian transformer-based language model which outperformed Multilingual BERT in the NER task, and Masala et al.[41] who created RoBERT using random texts crawled from the internet and formal texts from Romanian Wikipedia pages.

We want to develop 8 distinct RoBERTa variants in total and the motivation behind this is based on the linguistic diversity and complexity of Romanian as well as the varying preprocessing steps that might be needed in some NLP applications.

BERTweetRO model variants:

- Raw Cased
- Raw Uncased
- PreProcessed (PP) Cased
- PreProcessed (PP) Uncased
- Min Tokens Raw Cased
- Min Tokens Raw Uncased
- Min Tokens PP Cased
- Min Tokens PP Uncased

The first 4 variants (Raw Cased, Raw Uncased, PP Cased, and PP Uncased) differ from one another in the preprocessing steps and text case handling. Raw Cased preserves the original casing, Raw Uncased converts all characters to lowercase while the PP Cased and PP Uncased variants transform the data by removing all the URLs, Twitter mentions and hashtags, emoticons, and platform reserved keywords. These are the main contenders for our experiments that will allow us to see what impact (if any) case sensitivity and preprocessing has on the models. The next 4 variants (Min Tokens Raw Cased, Min Tokens Raw Uncased, Min Tokens PP Cased, and Min Tokens PP Uncased) are similar to the first ones, the difference being that in these cases we exclude the tweets that have less than 5 tokens from the dataset. With this filtration we want to remove as many noisy instances as possible from the training set in the hope of increasing the predictive power of the models.

Tokenization can be seen as the bridge that connects the natural representation of the texts used as inputs and the numerical values that encode the information such that it can be used by ML algorithms, and many different strategies to achieve this exist.

The Byte Pair Encoding (BPE) algorithm transforms texts into a tabular form and it's commonly used in various modeling tasks. A modification to the original algorithm was made allowing it to combine tokens that encode both single characters and full words [14]. In this case, all unique characters are considered to be an initial set of 1-character long n-grams. Next, the most common adjacent pairs of characters are merged to generate new 2-character long n-grams and all instances of previous pairs are replaced by this new token. This process repeats until a vocabulary of a predetermined size is reached. This version of BPE is very often set as the encoding method of LLMs and transformers. In contrast, the standard BPE doesn't merge the most frequent pair of bytes of data but instead replaces them with a new byte that was not seen in the initial dataset [57].

Due to the popularity and effectiveness of BPE we decided to apply it in our work with the help of the ByteLevelBPETokenizer implementation from Hugging Face[5] library. To train each variant of BERTweetRO Tokenizer we selected the following parameter configuration: (i) Vocabulary size of 16,000 tokens (ii) Minimum frequency threshold of 2 (iii) A set of special tokens containing $< s >$, $< pad >$, $< /s >$, $< unk >$, and $< mask >$.

The creation of the tokenizers consists in training them to transform the corpus of Romanian tweets in a number of ways that matches our target data variants. During this process the BPE algorithm discovers and learns statistical patterns based on the input texts and iteratively updates its vocabulary to capture as much information as possible for each subword unit. The resulting 8 tokenizers models were then saved for future usage.

To successfully learn our BERTweetRO models for Romanian NLP we selected an internal configuration that can yield good performances in relation to the training times and we integrated the previously trained tokenizers with each BERTweetRO variant in a consistent way to ensure that the hyperparameters and the end-to-end system allows for a fair comparison of performances in the downstream tasks.

We decided to employ the approach called Masked Language Modeling (MLM), implemented with the help of Hugging's RobertaForMaskedLM, which is a pretraining technique that enables transformers to predict masked tokens from input sequences. This is done without the need for labeled data making it an unsupervised learning method and unlike other traditional algorithms, that can only predict the next token in a given sequence, MLM can use both the previous and following tokens to predict a masked one. The architectural specifications of our BERTweetRO models are as follows: (i) Hidden size of 768 (ii) 12 attention heads (iii) 12 hidden layer (iv) MLM probability of 15%.

The BERTweetRO variants were trained over 5 epochs as we observed that it's sufficient to lead to an acceptable level of convergence without costing too much in terms of execution time. The choice of a MLM probability of 15% aligns with literature recommendations [38, 30], based on the reasoning that models can't learn good representations when too much or too little text is masked. Pretraining was done on our GPU with a batch size of 16 and the total execution time for all 8 variants was a little under 4 hours, which is decent if we consider the high computational overhead that is expected when creating transformers from scratch.

To fine tune for SA new layers need to be added on top of a pretrained BERT or RoBERTa model after which the entire architecture is trained in a supervised fashion on a annotated dataset. This allows the model to find and learn sentiment related features

[5] `https://huggingface.co/`

from the data such that it can make predictions on never seen before texts based. The quality of the model depends, among others, on the volume of data, the optimization process, and the number of iterations.

When talking about underrepresented languages, such as Romanian, we mention the work of Ciobotaru et al. [17] in which the authors trained a fastText-based model and fine-tuned a standard BERT-based model then compared their performances. They selected a public dataset containing COVID-19 related Twitter posts split in 2 categories (negative and positive). Next, they built upon this dataset by adding the "neutral" sentiment class and by adding more text samples to all classes. This new dataset was used as the benchmark in their experiments and the reported results on the test set showed that BERT achieved a F1 macro score of 0.84 while fastText had a worse score of only 0.73.

As stated in Chapter 3, we couldn't find a Romanian dataset fit for SA of microblogging content so we translated the Twitter US Airline Sentiment Tweets dataset to Romanian. Next we organized a series of experiments using the newly translated Romanian dataset together with all of our 8 BERTweetRO MLM variants. With Hugging Face's BertClassifier we fine-tuned each variant for the SA task using the correct tokenizer. Depending on each model variant, the associated preprocessing pipeline was executed in the same manner it was used during pre-training in order to ensure that a valid comparison between the models can be carried out in the future.

We'll evaluate the performance of our 8 sentiment-tuned BERTweetRO variants in a comparative study in which we include the best classifiers from Chapter 3. We note here that all the traditional sentiment classifiers from the comparative study underwent a rigorous hyperparameter optimization process. Thus, it is important to highlight that the fine tuned models will be compared against classifiers that have achieved peak performance given the selected dataset.

For our BERTweetRO variants we added an extra sequential layer for classification that can handle the output of the pretrained layers and the expected sentiment labels. As in the case of Multilingual BERT, our variants were not subjected to the hyperparameter optimization process due to time constraints. Instead, the fine-tuning parameters were chosen based on standard industry recommendations but also by considering the initial parameters that were used to pretrain the models: batch size of 32, BERT hidden size of 768, classification hidden size of 75, a max token length of 80, ReLU as the activation function, and categorical cross-entropy as the loss function. The number of epochs, ranging from 2 to 10, that offers a decent level of accuracy was investigated and identified individually for each BERTweetRO variant as well as for Multilingual BERT.

The results of our comparative analysis are listed in Table 5.2 in which the models are evaluated strictly on the test set. Given the unbalanced nature of our data we set Macro F1 as the main measure of predictive performance. By doing this we want to make sure that our evaluation treats each class in an equal manner in order to offer a correct interpretation of model effectiveness across all sentiments. Therefore we filtered the table based on the Macro-F1 scores in descending order which means that the best results are presented at the top.

We can see that Multilingual BERT outperformed all the other classifiers by having higher scores across all considered metrics but it's important to note that our best performing variants, namely BERTweetRO Raw Cased and BERTweetRO Raw Uncased, achieved a similar performance. The differences between them and M-BERT are fairly small with Macro F1 around 3% lower and the other 2 metrics around 2% lower. This outcome was somewhat expected if we take into account the difference in the size of data

Classifier	Encoding	Macro F1	Weighted F1	Accuracy
Multilingual BERT	Multilingual Tokenizer	74.81	80.50	80.99
BERTweetRO Raw Cased	BERTweetRO Tokenizer Raw Cased	72.11	78.40	78.74
BERTweetRO Raw Uncased	BERTweetRO Tokenizer Raw Uncased	72.07	78.33	78.66
Bernoulli NB	TFIDF	71.91	78.20	78.20
BERTweetRO Raw Min Tokens Cased	BERTweetRO Tokenizer Raw Min Tokens Cased	71.67	78.14	78.61
BERTweetRO Raw Min Tokens Uncased	BERTweetRO Tokenizer Raw Min Tokens Uncased	71.58	78.00	78.47
LSTM	Word2Vec	71.39	77.98	78.17
Linear SVM	TFIDF	70.54	77.47	78.36
DNN	Word2Vec	69.19	76.23	77.20
Logistic Regression	TFIDF	69.04	76.45	77.81
CNN	Word2Vec	68.67	76.00	77.69
BERTweetRO PP Min Tokens Cased	BERTweetRO Tokenizer PP Min Tokens Cased	64.21	73.10	72.86
BERTweetRO PP Cased	BERTweetRO Tokenizer PP Cased	43.84	59.40	64.50
BERTweetRO PP Uncased	BERTweetRO Tokenizer PP Uncased	42.35	58.73	64.01
Random Forest	TFIDF	38.17	54.71	65.20
BERTweetRO PP Min Tokens Uncased	BERTweetRO Tokenizer PP Min Tokens Uncased	25.62	47.98	62.42

Table 5.2: Sentiment Analysis performance

used for pre-training, as M-BERT benefited from a much larger volume and diverse data whereas our variants were trained on a considerable smaller dataset (≈51,000 tweets).

Surprisingly, the BERTweetRO variants that were trained on texts with a minimum token constraint (BERTweetRO Raw Min Tokens Cased and Uncased) also had competitive results. This means that by limiting the data used to train the models, i.e. keeping only the texts with more than 5 tokens, the predictive performance that can be achieved is not reduced in a significant manner and additionally this can improve to some degree the execution speeds. Bernoulli NB, despite its simplicity, obtained good results placing it in between these 4 variants. With this exception the BERTWeetRO models that did not use the text preprocessing pipeline performed better than all the classic and deep learning models.

Another point worth highlighting is that regardless of the BERTweetRO variant, those trained on original text casing (the Cased variants) have marginally better results than their counterparts trained on lowercased data (the Uncased variants). In the middle of the ranking we have LSTM, Linear SVM, DNN, Logistic Regression, and CNN with decent performances which makes them suitable for usage in real scenarios.

At the bottom of the table, where the models with the worst results are placed, we have all the BERTweetRO variants that were paired with our custom text preprocessing module which clearly shows that this process negatively affected their predictive performances. This means that better BERT-based models can be developed by simply pretraining and fine tuning them on raw social media data without the need for additional text cleaning or feature engineering. These models together with Random Forest had by far the lowest predictive performance meaning that they cannot be considered for SA.

Fine tuning for TC involves the use of an existing pre-trained model and adapting it to classify the inputed texts based on the discussion topics that they convey. This is achieved by incorporating task specific layers in the BERT or RoBERTa model after which the entire architecture is trained in a supervised fashion on a annotated dataset. This allows the model to find and learn topic related features from the data and to make predictions on never seen texts with the help of these learned representations.

Next, similar to SA fine-tuning, we ran a number of experiments using the translated News Category dataset together with all 8 variants of our pretrained BERTweetRO MLM models. We'll evaluate the performance of our 8 topic-tuned BERTweetRO variants in another comparative study in which we include the best classifiers from Chapter 4.

For our BERTweetRO variants we added an extra sequential layer for classification on top of the existing architecture that can handle the output of the pre-trained layers and the expected topic labels. As in the case of Multilingual BERT, our variants were not subjected to the hyperparameter optimization process due to time constraints. Instead, the parameters used for fine-tuning were chosen based on standard industry recommendations but also by considering the initial parameters that were set to pretrain the models: batch size of 32, BERT hidden size of 768, classification hidden size of 128, a max token length of 120, ReLU as the activation function, and categorical cross-entropy as the loss function. The number of epochs, ranging from 2 to 10, which leads to an acceptable level of accuracy was investigated and identified individually for each BERTweetRO variant as well as for M-BERT.

Unlike SA, where the goal is to detect a text's global polarity, the difficulty of TC resides also in the big number of target classes which often overlap [25, 39]. To overcome this, some authors [26, 52] use the Top-K accuracy instead of the standard one. Rather than classifying a text into a single class and comparing it to the a-priori label, the model will predict the K most likely classes and if the correct label is among them, we consider the text as being correctly classified. We take this into account and report the standard accuracy (i.e. Top-1), as well as Top-2 and Top-3, measured strictly on the test set.

In Table 5.3 we list the results of our comparative study, filtered using the Top-1 accuracy in descending order meaning that the best models appear at the beginning. Here we can see Multilingual BERT in the first place with impressive Top-1, Top-2, and Top-3 accuracies of 72.63%, 85.56%, and 90.25%. This result was expected if we consider the huge volume of data on which this transformer model was pretrained, allowing it to generate robust initial representations, even in the Romanian language, which are then easily adjusted for TC with the help of our translated dataset.

In the race for the runner up position we have several models with similar scores across all 3 evaluation metrics, namely Linear SVM and the 4 Raw BERTweetRO variants. These classifiers reached Top-1 accuracies between $\approx 65\%$ and $\approx 66\%$, which are good enough for real life applications. As in the case of SA fine-tuning, the BERTweetRO variants that didn't use the custom text preprocessing pipeline obtained better results than the other variants that did use it. The difference between our best variants and Multilingual BERT is around 6% but can be considered decent if we take into account the relative small size of the data we used for pretraining (around 51,000 texts). This means that by employing a richer dataset and by applying hyperparameter optimization we could potentially enhance the performance of the BERTweetRO variants in future iterations.

BERTweetRO Raw Min Tokens Uncased/Cased also have competitive performances, as they did for SA, which reconfirms that by restricting the data used for fine tuning the predictive performance that can be achieved is not downgraded while training times are

Classifier	Encoding	Top-1 Acc.	Top-2 Acc.	Top-3 Acc.	opt (s)	train (s)	test (s)
Multilingual BERT	Multilingual Tokenizer	72.63	85.56	90.25	N/A	7498	157
Linear SVM	TFIDF	66.73	80.05	85.30	11803	45.91	0.042
BERTweetRO Raw Uncased	BERTweetRO Tokenizer Raw Uncased	66.14	79.21	84.93	N/A	8436	135
BERTweetRO Raw Min Tokens Uncased	BERTweetRO Tokenizer Raw Min Tokens Uncased	66.07	79.10	84.80	N/A	6899	157
BERTweetRO Raw Min Tokens Cased	BERTweetRO Tokenizer Raw Min Tokens Cased	65.78	79.02	84.79	N/A	6923	157
BERTweetRO Raw Cased	BERTweetRO Tokenizer Raw Cased	65.63	78.75	84.48		8442	132
Bernoulli NB	TFIDF	62.80	77.70	84.11	398	0.59	0.04
CNN	Word2Vec	61.66	74.05	79.28	36797	56.98	1.65
BERTweetRO PP Min Tokens Cased	BERTweetRO Tokenizer PP Min Tokens Cased	54.55	66.60	72.94	N/A	6046	137
LSTM	Word2Vec	53.50	65.59	72.39	63605	119.1	6.16
Random Forest	TFIDF	16.56	28.89	37	845	0.6	0.183
BERTweetRO PP Min Tokens Uncased	BERTweetRO Tokenizer PP Min Tokens Uncased	16.56	28.89	37	N/A	6019	135
BERTweetRO PP Cased	BERTweetRO Tokenizer PP Cased	16.56	28.89	37	N/A	6027	135
BERTweetRO PP Uncased	BERTweetRO Tokenizer PP Uncased	16.56	28.89	37	N/A	6022	135

Table 5.3: Topic Classification performance

improved. This is evident when we compare the training times between the Raw and Min Tokens variants and see that the Min Tokens variants were around 22% faster.

Bernoulli NB and CNN are next, with slightly lower scores, both having a similar performance when talking about Top-1 but in the case of Top-2 and Top-3 CNN lags behind by a pretty noticeable margin. These models should behave more or less the same when predicting the main topic of a text, but Bernoulli NB should be preferred if one considers the second and third most probable topics as being important to their use case.

BERTweetRO PP Min Tokens Cased and LSTM share fourth place in our ranking with modest results across all the considered metrics. At the bottom of the table we have RF and the 4 BERTweetRO variants that incorporated the text preprocessing module, all of them having by far the worst predictive performance, achieving a Top-1 accuracy of only 16.5%. This once again underscores the effectiveness of simply pretraining and fine tuning BERT models on raw social media texts without the need for any text cleaning or feature engineering. Hence, this group of models are not viable for TC in real life applications.

Chapter 6

Assessing Sentiment Analysis Performance on Real Cases

Given that the final purpose of our work is to apply the learned models for inferring the polarity of any Romanian tweet, we manually labeled two small test sets, each one containing 120 distinct tweets. The first one includes tweets specific to the airline industry, comparable with the ones used for training our models, and the second one includes general tweets. We will evaluate on these test sets the best performing models as reported in Chapter 5: Multilingual BERT, BERTweetRO Raw Uncased, Bernoulli NB, LSTM, and DNN [47]. Additionally, we'll compare these models against a public third-party sentiment analysis tool for Romanian called Sentimetric[1] to see where we stand in relation to a commercially available solution.

The tweets were manually labeled by 5 human volunteers who were instructed in advance on how this process should be carried out. Each volunteer expressed an opinion about the polarity of the tweet and the final sentiment was set as the one selected by the majority. Labeling statistics regarding how they assessed the polarity is presented in Table 6.1. We shall note that the labeling task seemed to be a difficult one even for the volunteers, as for only 43 tweets (35.8%) in the case of airline industry specific dataset and 47 tweets (39.2%) in the case of general tweets all of the 5 contributors reached a unanimous decision. Furthermore, the class distribution of these tweets is significantly different from that of the Twitter US Airline Sentiment Tweets.

Dataset	Negative	Neutral	Positive	Unanimous Annotation
Airline industry-specific tweets	51 (46.5%)	36 (30%)	33 (27.5%)	43 (35.8%)
General tweets	45 (37.5%)	32 (26.66%)	43 (35.8%)	47 (39.2%)
Twitter US Airline Sentiment Tweets	63%	21%	16%	N/A

Table 6.1: Manual sentiment labeling statistics (tweet number and percentage)

As in the case of the fine tuning experiments we report Macro F1, Weighted F1, and Accuracy as the evaluation metrics for each classifier but seeing the imbalanced distribution of labels of both dateset we again have to set Macro F1 as the main measure of model performance.

Table 6.2 contains the predictive performance of our target models on the 120 real

[1] `http://sentimetric.ro/`

life Romanian tweets that relate to the aviation industry. In this case we can see that Bernoulli Naive Bayes (NB) achieved the highest Macro F1 score of 61.18% and in second place, with a marginally lower score, we have Multilingual BERT. This result is a little surprising considering that Multilingual BERT had better results on the evaluation set used in the fine tuning experiments of Chapter 5 but the success of NB could be attributed to the hyperparameter optimization process that it went through.

Classifier	Encoding	Macro F1	Weighted F1	Accuracy
Bernoulli NB	TFIDF	61.18	63.11	65
Multilingual BERT	Multilingual Tokenizer	60.45	63.38	65.83
BERTweetRO Raw Uncased	BERTweetRO Tokenizer Raw Uncased	54.57	56.68	60
LSTM	Word2Vec	52.71	55.18	58.33
DNN	Word2Vec	52.22	54.9	59.17
Sentimetric	N/A	45.72	46.99	47.5

Table 6.2: Model performances on Romanian aviation industry-specific tweets

The dataset has a small number of samples but despite this our BERTweetRO Raw Uncased variant managed to secure an honorable third place across all evaluation metrics. Even though that it failed to surpass Bernoulli NB and Multilingual BERT, BERTweetRO's performance is better than the deep learning LSTM and DNN models. The Macro F1 of 54.5%, which is around 6% lower than the best score, can be considered acceptable given that the human volunteers also had difficulties when labeling the texts.

The most important thing that we want to highlight here is that all of our models outperformed Sentimetric. This shows the positive impact of using a custom methodology for training and validating ML models when compared to off-the-shelf solutions. This also confirms the value of domain specific knowledge for getting better results in such contexts, as we created our models with tweets from the same domain.

In Table 6.3 we present model performances on the Romanian general tweets dataset. In this case things are a little different as Multilingual BERT achieved the best result with a Macro F1 of 55.22%, followed closely by BERTweetRO Raw Uncased with a negligible difference in score of only 1%. In both this assessment and the previous one, the Transformer-based models placed at the top which means that they're more reliable for sentiment prediction in practice.

Classifier	Encoding	Macro F1	Weighted F1	Accuracy
Multilingual BERT	Multilingual Tokenizer	52.22	54.17	55.85
BERTweetRO Raw Uncased	BERTweetRO Tokenizer Raw Uncased	51.35	52.39	54.17
Bernoulli NB	TFIDF	48.48	49.42	48.33
DNN	Word2Vec	48.16	49.29	50.83
Sentimetric		46.16	47.3	49.17
LSTM	Word2Vec	43.17	44.29	45.83

Table 6.3: Model performances on Romanian general tweets

On the other hand, Bernoulli NB and DNN have more modest results that place them in the middle of the ranking but more surprising is that LSTM delivered a significantly worse performance in this case, being behind all other models including Sentimetric. The reasons for why this happened requires future investigations but it's possible that the

complexity of LSTM together with its sensitivity to the shape of input data could have affected its ability to correctly recognize the sentiment patterns from these samples.

An interesting detail we want to point out is that the overall performance of the models on these general tweets is lower than the performance on aviation industry tweets. This decline, which is more obvious for the classic and deep learning models, is a direct result of domain differences between the texts used for training and the ones used for evaluation. For Multilingual BERT and BERTweetRO the decline is less serious due to the fact that they were pretrained on varied texts, and thus managed to better adapt in this scenario.

Nonetheless, as in the case of the first evaluation, we note that all our models (with the exception of LSTM) have significantly outperformed the commercial solution selected as the benchmark for comparison. This once again validates the importance of custom fine tuning and model optimization in achieving superior results for Romanian SA.

For both domains, our models' results are lower than those obtained on the translated test set used in Chapter 5, because now the tweets are real ones, not translated, and their inherent characteristics differ, i.e. from a statistical point of view the sets are extracted from different statistical populations.

Chapter 7

Conclusion

With this chapter we finalize our doctoral thesis by going over and summarizing the most important elements presented in this research. At the heart of our motivation for undertaking this study we have social media platforms. These evolved from the "social networks" that appeared in the late 90's and became popular by mid 2000s. Initially, social networks were nothing more than basic websites offering limited functionalities, mainly related to the distribution of photos and messaging between explicitly connected users. Even so, their user base increased exponentially very fast and over time the modern social media platforms emerged. These platforms now offer an incredible amount of complex and interconnected features, among which we mention: posting content to global audiences, sharing and consuming any type of content (texts, images, videos, advertisements, etc.), and easy access at all times from mobile or desktop devices.

Thus, it comes as no surprise that 68% of the global population is currently active on social media platforms and for Romania this percentange is even higher, with 90% of the country's total population being active on such platforms by the start of 2025. These numbers were another motivating factor for us because the users generate large volumes of textual data, rich in variety, with high velocity (i.e. big data characteristics). Therefore, it's obvious that both researchers and public or private entities can benefit greatly from the information hidden in this sort of data.

However, extracting useful information from social media texts is problematic. As opposed to more literary texts, microblogging texts are filled with "bad language" elements, such as: mistakes in grammar, typos, non-standard abbreviations, the use of emojis. In addition to this, they are also informal by nature and short due to the size limitations imposed by the platform, for example a Twitter tweet has a limit of 280 characters, a TikTok comment has a limit of 150. For these reasons, the rule or dictionary based approaches for NLP fail most of the time and the literature strongly recommends the usage of ML to overcome these issues. For popular languages like English, French, or Spanish there are plenty resources and tools that can deal with social media content, but for Romanian (and other under-represented languages) the situation is more challenging because they're are much harder to find and employ in production environments. So, contributing to the existing body of Romanian NLP resources is another motivating factor for us.

At the beginning of this thesis we set a number of objectives, the first one being the study of Sentiment Analysis for Romanian social media texts. After reviewing the literature, we discovered that most research works treat SA as a binary classification task (negative vs. positive). This approach is good enough if we know beforehand that the texts to be analyzed are very polarizing but in most scenarios this is not the case.

Additionally, a lot of works focus on other types of text, i.e. online product reviews.

We searched in many online repositories for a labeled datased that could satisfy our research needs but unfortunately we couldn't find one. Research works for other languages suggest that translation can be used as a means of creating new and usable learning resources so we decided to test this hypothesis for Romanian. Hence, we selected the Twitter US Airline Sentiment Tweets dataset for our experiments because it contains many tweets, manually labeled with 3 sentiment classes (negative, neutral, positive). We then translated this English dataset to Romanian using an automated translation service in order to create our training set. Using this data we trained and evaluated a wide selection of ML algorithms from the area of classic and deep learning, plus the popular Multilingual BERT as an example of Transformer model.

The results of SA experiments show that the Romanian models are on par with the equivalent English counterparts, the variation in accuracy between them being only around ±2%. In this context we set the state-of-the-art performance for multinomial sentiment classification with M-BERT at 83% for English and 81% for Romanian. Two of the classic learners, Bernoulli NB and Linear SVM, showed competitive performances with M-BERT by achieving 78% accuracy in both languages.

Our second objective is dedicated to identifying discussion topics from short form texts. To achieve this goal we decided to employ supervised Topic Classification instead of Topic Modeling, due to the drawbacks that come with the unsupervised nature of TM methods: data instability, finding the optimal number of topics to extract is not trivial, and the extracted sets of topic keywords require human intervention to be annotated with relevant topic labels. Compared to SA, the current research efforts in the area of TC for Romanian microblogging content is even more limited. We could only find 1 study which utilized the TC approach but their dataset is small and not publicly available.

Seeing the encouraging results of our SA experiments, we decided again to find a reliable English dataset and translate it to Romanian. For this we selected the News Category Dataset which contains over 200,000 blog news headlines and descriptions, grouped in 41 topic classes. We discovered that some of the original topics were highly specific while others were overlapping so, as a next step, we improved this dataset by clustering together granular and synonymous categories in order to create 26 truly distinct topic labels. Next we trained and tested a number of ML models on both sets of data. Our findings here show that the Romanian models tend to have slightly lower scores than the English models, but for the best performing classifiers this difference in negligible (with decreases of 2–3%). Due to potential topic overlap in single text instances we report Top-1, Top-2, and Top-3 accuracy as the evaluation metric. As such, M-BERT once again sets the state-of-the-art with 74.85% Top-1, 87.29% Top-2, and 91.73% Top-3 on the English data, and 72.63% Top-1, 85.56% Top-2, and 90.25% Top-3 on the Romanian data. Linear SVM is second with 68% Top-1, 82% Top-2, and 87% Top-3 for English and 67% Top-1, 80% Top-2, and 85% Top-3 for Romanian.

All our SA and TC models, except BERT, were subjected to a hyperparameter optimization process, implemented with evolutionary algorithms, in order to achieve the highest predictive performance possible. It is worth mentioning that we also compared the execution times of the models during optimization, training, and inference. Thus, with our experimental findings at hand, others can more easily select the model that suits their needs based on the expected predictive performance in relation to hardware usage.

Our third objective refers to the creation of Transformer models designed for Romanian social media texts. Using the Twitter Stream Archive, which contains public tweets

scrapped over a long period of time, we selected a 12 month interval for our study. After investigating a random sample of instances from this interval, we found that most of the language field metadata for the posts pre-labeled as Romanian were in fact misclassifications. With respect to this we implemented a preprocessing module to clean the texts, ran our own language identification process on them, and in the end managed to extract around 51,000 true Romanian posts. Using this corpus we pre-trained from scratch 8 variants of BERTweetRO models, based on the RoBERTa architecture with MLM. These variants differ between each other in the text case sensitivity (cased vs. uncased), text format (raw vs. preprocessed), and minimum number of tokens in the texts used for pre-training (no min vs. 5 min tokens).

Next, we fine-tuned our variants on the Romanian translated SA and TC datasets, and conducted a new series of experiments. Here we discovered that the variants using the raw texts are the best, but the ones with min token constraints have only a slightly lower predictive performance while being faster to run. Text preprocessing had a major negative impact on predictive capability and we don't recommend pairing it with transformers. For SA, the best BERTweetRO variants placed second after M-BERT, with a Macro F1 of $\approx$ 2.5% points lower than M-BERT's score of 74.8%. For TC, the best BERTweetRO variants placed third after Linear SVM but their accuracies are almost identical to Linear SVM's. These results are quite impressive if we consider the extreme difference in size between the data we used to fine-tune BERTweetRO against Multilingual BERT's extensive data.

Our final objective was set on comparing our best performing classifiers which include BERTweetRO, M-BERT, Linear SVM, Bernoulli NB, and DNN against a Romanian commercial SA classifier called Sentimetric. In order to truly validate our translation methodology we collected and manually labeled with sentiment 2 sample sets of real-life Romanian tweets: one containing tweets related to the airline industry, the other containing general tweets. In both domains, all of our classifiers surpassed Sentimetric, therefore validating our end-to-end modeling framework.

With the scenarios and experiments covered in this thesis we prove, without a doubt, that automated translation from English to Romanian is a viable alternative for creating useful NLP resources. Moreover, we show that it's possible to create BERT-based models from scratch using a relatively small pretraining dataset of native Romanian texts, fine-tune these models for other downstream tasks using texts translated to Romanian, and achieve good results in this process. Thus, we consider that our work brings valuable contributions to the development of linguistic resources for processing short Romanian texts. Also, these finding should be considered by other researchers that work with under-represented languages and face similar struggles due to limited open-source materials.

Bibliography

[1] Charu C Aggarwal and Charu C Aggarwal. *Mining text data*. Springer, 2015.

[2] Amritanshu Agrawal, Wei Fu, and Tim Menzies. What is wrong with topic modeling? and how to fix it using search-based software engineering. *Information and Software Technology*, 98:74–88, 2018.

[3] Milam Aiken. An updated evaluation of google translate accuracy. *Studies in Linguistics and Literature*, 3:p253, 2019.

[4] Federico Albanese and Esteban Feuerstein. Improved topic modeling in twitter through community pooling. In *String Processing and Information Retrieval - 28th Intl. Symposium, SPIRE 2021*, volume 12944 of *LNCS*, pages 209–216. Springer, 2021.

[5] Andrei-Marius Avram, Darius Catrina, Dumitru-Clementin Cercel, Mihai Dascalu, Traian Rebedea, Vasile Florian Pais, and Dan Tufis. Distilling the knowledge of romanian berts using multiple teachers. *CoRR*, abs/2112.12650, 2021.

[6] Alexandra Balahur and José Manuel Perea Ortega. Sentiment analysis system adaptation for multilingual processing: The case of tweets. *Information Processing and Management*, 51(4):547–556, 2015.

[7] Alexandra Balahur and Marco Turchi. Comparative experiments using supervised learning and machine translation for multilingual sentiment analysis. *Computer Speech and Language*, 28(1):56–75, 2014.

[8] Carmen Banea, Rada Mihalcea, and Janyce Wiebe. Multilingual subjectivity: Are more languages better? In Chu-Ren Huang and Dan Jurafsky, editors, *COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings*, pages 28–36. Tsinghua University Press, 2010.

[9] Francesco Barbieri, Luis Espinosa Anke, and José Camacho-Collados. XLM-T: A multilingual language model toolkit for twitter. *CoRR*, abs/2104.12250, 2021.

[10] Valentin Barrière and Alexandra Balahur. Improving sentiment analysis over non-english tweets using multilingual transformers and automatic translation for data-augmentation. In Donia Scott, Núria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020*, pages 266–271. International Committee on Computational Linguistics, 2020.

[11] David Blei and John Lafferty. Correlated topic models. *Advances in neural information processing systems*, 18:147, 2006.

[12] Jordan L. Boyd-Graber and David M. Blei. Syntactic topic models. In *Proc. of the 22nd Annual Conf. on Neural Information Processing Systems, 2008*, pages 185–192, 2008.

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

[14] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[15] Xueqi Cheng, Xiaohui Yan, Yanyan Lan, and Jiafeng Guo. Btm: Topic modeling over short texts. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):2928–2941, 2014.

[16] Alexandra Ciobotaru and Liviu P. Dinu. RED: A novel dataset for romanian emotion detection from tweets. In Galia Angelova, Maria Kunilovskaya, Ruslan Mitkov, and Ivelina Nikolova-Koleva, editors, *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), Held Online, 1-3September, 2021*, pages 291–300. INCOMA Ltd., 2021.

[17] Alexandra Ciobotaru and Liviu P Dinu. Sart & covidsentiro: Datasets for sentiment analysis applied to analyzing covid-19 vaccination perception in romanian tweets. *Procedia Computer Science*, 225:1331–1339, 2023.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, pages 4171–4186. Association for Computational Linguistics, 2019.

[19] Manuel Carlos Díaz-Galiano, Manuel García Vega, Edgar Casasola, Luis Chiruzzo, Miguel Ángel García Cumbreras, Eugenio Martínez Cámara, Daniela Moctezuma, Arturo Montejo-Ráez, Marco Antonio Sobrevilla Cabezudo, Eric Sadit Tellez, et al. Overview of tass 2019: One more further for the global spanish sentiment analysis corpus. In *IberLEF@ SEPLN*, pages 550–560, 2019.

[20] Susan T Dumais. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230, 2004.

[21] Stefan Daniel Dumitrescu, Andrei-Marius Avram, and Sampo Pyysalo. The birth of romanian bert. *arXiv preprint arXiv:2009.08712*, 2020.

[22] Stefan Daniel Dumitrescu, Petru Rebeja, Beáta Lorincz, Mihaela Gaman, Andrei-Marius Avram, Mihai Ilie, Andrei Pruteanu, Adriana Stan, Lorena Rosia, Cristina Iacobescu, Luciana Morogan, George Dima, Gabriel Marchidan, Traian Rebedea, Madalina Chitez, Dani Yogatama, Sebastian Ruder, Radu Tudor Ionescu, Razvan Pascanu, and Viorica Patraucean. Liro: Benchmark and leaderboard for romanian language tasks. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021.

[23] Jacob Eisenstein. What to do about bad language on the internet. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, 2013*, pages 359–369. The Association for Computational Linguistics, 2013.

[24] Ahmed Elgohary, Matthias Boehm, Peter J Haas, Frederick R Reiss, and Berthold Reinwald. Compressed linear algebra for large-scale machine learning. *Proceedings of the VLDB Endowment*, 9(12):960–971, 2016.

[25] Matthew Gentzkow, Bryan Kelly, and Matt Taddy. Text as data. *Journal of Economic Literature*, 57(3):535–574, 2019.

[26] Maya R. Gupta, Samy Bengio, and Jason Weston. Training highly multiclass classifiers. *Journal of Machine Learning Research*, 15(1):1461–1492, 2014.

[27] Felix Hamborg, Karsten Donnay, Paola Merlo, et al. Newsmtsc: a dataset for (multi-) target-dependent sentiment classification in political news articles. Association for Computational Linguistics (ACL), 2021.

[28] Liangjie Hong and Brian D. Davison. Empirical study of topic modeling in Twitter. In *3rd Workshop on Social Network Mining and Analysis, SNAKDD 2009* , pages 80–88. ACM, 2010.

[29] Lucian Istrati and Alexandra Ciobotaru. Automatic monitoring and analysis of brands using data extracted from twitter in romanian. In Kohei Arai, editor, *Intelligent Systems and Applications - Proceedings of the 2021 Intelligent Systems Conference, IntelliSys 2021, Amsterdam, The Netherlands, 2-3 September, 2021, Volume 3*, volume 296 of *Lecture Notes in Networks and Systems*, pages 55–75. Springer, 2021.

[30] Peter Izsak, Moshe Berchansky, and Omer Levy. How to train bert with an academic budget. *arXiv preprint arXiv:2104.07705*, 2021.

[31] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Reading text in the wild with convolutional neural networks. *International journal of computer vision*, 116(1):1–20, 2016.

[32] Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142. Springer Verlag, 1998.

[33] Ian Jolliffe and Jorge Cadima. Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374:20150202, 04 2016.

[34] Daniel Jurafsky and James H Martin. Vector semantics and embeddings. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, pages 270–85, 2019.

[35] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. Text classification algorithms: A survey. *Information*, 10(4):150, 2019.

[36] Petra Kralj Novak, Jasmina Smailović, Borut Sluban, and Igor Mozetič. Sentiment of emojis. *Plos One*, 10(12):1–22, 12 2015.

[37] Quoc V. Le and Tomás Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1188–1196. JMLR.org, 2014.

[38] Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. Pmi-masking: Principled masking of correlated spans. *arXiv preprint arXiv:2010.01825*, 2020.

[39] Bing Liu. *Sentiment analysis: Mining opinions, sentiments, and emotions*. Cambridge university press, 2020.

[40] Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. FastBERT: a self-distilling BERT with adaptive inference time. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 6035–6044. Association for Computational Linguistics, 2020.

[41] Mihai Masala, Stefan Ruseti, and Mihai Dascalu. Robert–a romanian bert model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6626–6637, 2020.

[42] Andrew McCallum and Kamal Nigam. A comparison of event models for naive bayes text classification. In *Learning for Text Categorization: Papers from the 1998 AAAI Workshop*, pages 41–48, 1998.

[43] Walaa Medhat, Ahmed Hassan, and Hoda Korashy. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, 5(4):1093–1113, 2014.

[44] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings*, pages 3111–3119, 2013.

[45] Saif Mohammad, Cody Dunne, and Bonnie Dorr. Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 599–608, 2009.

[46] Naoki Mori, Masayuki Takeda, and Keinosuke Matsumoto. A comparison study between genetic algorithms and bayesian optimize algorithms by novel indices. In Hans-Georg Beyer and Una-May O'Reilly, editors, *Genetic and Evolutionary Computation Conference, GECCO 2005, Proceedings, Washington DC, USA, June 25-29, 2005*, pages 1485–1492. ACM, 2005.

[47] Dan Claudiu Neagu. Bertweetro: pre-trained language models for romanian social media content. *Studia Universitatis Babes-Bolyai Oeconomica*, 2025.

[48] Dan Claudiu Neagu, Andrei Bogdan Rus, Mihai Grec, Mihai Boroianu, and Gheorghe Cosmin Silaghi. *Topic Classification for Short Texts*, pages 207–222. Springer International Publishing, Cham, 2023.

[49] Dan Claudiu Neagu, Andrei Bogdan Rus, Mihai Grec, Mihai Augustin Boroianu, Nicolae Bogdan, and Attila Gal. Towards sentiment analysis for romanian twitter content. *Algorithms*, 15(10):357, 2022.

[50] Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani, editors, *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001]*, pages 841–848. MIT Press, 2001.

[51] Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. Bertweet: A pre-trained language model for english tweets. In Qun Liu and David Schlangen, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos*, pages 9–14. Association for Computational Linguistics, 2020.

[52] Sechan Oh. Top-k hierarchical classification. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 2450–2456. AAAI Press, 2017.

[53] Varun Kumar Ojha, Ajith Abraham, and Václav Snásel. Metaheuristic design of feedforward neural networks: A review of two decades of research. *Engineering Applications of Artificial Intelligence*, 60:97–116, 2017.

[54] Najiba Ouled Omar, Azza Harbaoui, and Henda Ben Ghezala. Opinion mining and sentiment analysis on deft. *International Journal of Cognitive and Language Sciences*, 15(1):54 – 57, 2021.

[55] Andrew Ortony, Gerald L Clore, and Allan Collins. *The cognitive structure of emotions*. Cambridge university press, 2022.

[56] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In Waleed Ammar, Annie Louis, and Nasrin Mostafazadeh, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Demonstrations*, pages 48–53. Association for Computational Linguistics, 2019.

[57] Gerhard Paaß and Sven Giesselbach. *Pre-trained Language Models*, pages 19–78. Springer International Publishing, Cham, 2023.

[58] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[59] Razvan Pascanu, Tomás Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1310–1318. JMLR.org, 2013.

[60] V. Paul Pauca, Farial Shahnaz, Michael W. Berry, and Robert J. Plemmons. Text mining using non-negative matrix factorizations. In Michael W. Berry, Umeshwar Dayal, Chandrika Kamath, and David B. Skillicorn, editors, *Proceedings of the Fourth SIAM International Conference on Data Mining, 2004*, pages 452–456. SIAM, 2004.

[61] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014.

[62] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual bert? In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 4996–5001. Association for Computational Linguistics, 2019.

[63] Marco Pota, Mirko Ventura, Hamido Fujita, and Massimo Esposito. Multilingual evaluation of pre-processing for bert-based sentiment analysis of tweets. *Expert Systems with Applications*, 181:115119, 2021.

[64] Radim Rehurek and Petr Sojka. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2), 2011.

[65] Sara Rosenthal, Noura Farra, and Preslav Nakov. Semeval-2017 task 4: Sentiment analysis in twitter. In Steven Bethard, Marine Carpuat, Marianna Apidianaki, Saif M. Mohammad, Daniel M. Cer, and David Jurgens, editors, *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 502–518. Association for Computational Linguistics, 2017.

[66] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing and management*, 24(5):513–523, 1988.

[67] Lucas Nunes Sequeira, Bruno Moreschi, Fábio Gagliardi Cozman, and Bernardo Fontes. An empirical accuracy law for sequential machine translation: the case of google translate. *CoRR*, abs/2003.02817, 2020.

[68] Ryan A Stevenson, Joseph A Mikels, and Thomas W James. Characterization of the affective norms for english words by discrete emotional categories. *Behavior research methods*, 39(4):1020–1024, 2007.

[69] Ayisha Tabassum and Rajendra R Patil. A survey on text pre-processing & feature extraction techniques in natural language processing. *International Research Journal of Engineering and Technology (IRJET)*, 7(06):4864–4867, 2020.

[70] Anca Maria Tache, Mihaela Gaman, and Radu Tudor Ionescu. Clustering word embeddings with self-organizing maps. application on laroseda - A large romanian sentiment data set. In Paola Merlo, Jörg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 949–956. Association for Computational Linguistics, 2021.

[71] Adrian Vasile, Roxana Rădulescu, and Ionel-Bujorel Păvăloiu. Topic classification in romanian blogosphere. In *12th Symposium on Neural Network Applications in Electrical Engineering (NEUREL)*, pages 131–134. IEEE, 2014.

[72] Ike Vayansky and Sathish AP Kumar. A review of topic modeling methods. *Information Systems*, 94:101582, 2020.

[73] Pradnya A Vikhar. Evolutionary algorithms: A critical review and its future prospects. In *2016 International conference on global trends in signal processing, information computing and communication (ICGTSPICC)*, pages 261–265. IEEE, 2016.

[74] Xuerui Wang and Andrew McCallum. Topics over time: a non-Markov continuous-time model of topical trends. In *12th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining, 2006*, pages 424–433. ACM, 2006.

[75] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019.

[76] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing [review article]. *IEEE Comput. Intell. Mag.*, 13(3):55–75, 2018.

[77] Jichuan Zeng, Jing Li, Yan Song, Cuiyun Gao, Michael R. Lyu, and Irwin King. Topic memory networks for short text classification. In *Proc. of the 2018 Conf. on Empirical Methods in Natural Language Processing*, pages 3120–3131. ACL, 2018.