Large Movie Review Dataset automatic sentiment analysis system

Prokhar Navitski
Potsdam University
Potsdam 2023

1. Introduction

This report presents the results of my work on the development of an automatic system for analysing sentiments using the example of the Large Movie Review Dataset[1].

The objective of the work was to develop a mechanism for automatic evaluation of text tonality. The stages of the work include training a language model on an example of texts with already defined tonality, followed by automatic analysis of texts without predetermined tonality. To improve the accuracy of my conclusions, I compare all the results obtained by training my model with the results of training the model of the database authors themselves.

2. Dataset

The core dataset consists of 50,000 reviews, evenly split into 25,000 for training and 25,000 for testing. It maintains a balanced distribution of labels (25,000 positive and 25,000 negative). An additional 50,000 unlabelled documents are provided for unsupervised learning.

To avoid correlation among reviews for the same movie, a maximum of 30 reviews per movie is enforced across the dataset. The training and testing sets are movie-disjoint to prevent performance gains from memorizing movie-specific terms.

In the labeled train/test sets, negative reviews have a score <= 4 out of 10, and positive reviews have a score >= 7 out of 10. Reviews with neutral ratings are excluded. In the unsupervised set, reviews of all ratings are included, maintaining an even distribution above and below a score of 5.

3. Tools

I used Python 3 programming language (version 3.9) and IDE PyChram 2023.1 to build an automatic model of text tone analysis.

The advantage of Python language for this task is that it has a large number of open source libraries, which are used for natural language processing and processing of the obtained data. In my project I used the following libraries:

[1] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning Word Vectors for Sentiment Analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (pp. 142-150). Association for Computational Linguistics. Retrieved from http://www.aclweb.org/anthology/P11-1015

spacy: spaCy is a natural language processing (NLP) library for Python. It provides pre-trained models for various languages and allows you to perform tasks like tokenization, part-of-speech tagging, named entity recognition, and more. It's known for its efficiency and ease of use in NLP tasks.

nltk (Natural Language Toolkit): NLTK is a comprehensive library for NLP in Python. It offers tools and resources for tasks such as text processing, tokenization, stemming, lemmatization, and text classification. It's widely used for research and education in NLP.

re (Regular Expressions): The "re" library in Python is used for working with regular expressions. Regular expressions are powerful patterns that allow you to search, match, and manipulate text data. It's often used for tasks like pattern matching, text extraction, and data validation.

unicodedata: The "unicodedata" module in Python provides access to the Unicode character database. It allows you to work with Unicode characters and perform operations like character normalization, classification, and property retrieval. It's useful when dealing with text data containing characters from different languages and character encodings.

Hardware:

OS	macOS Ventura 13.5.2
Processor	apple M1 chip
RAM	8 GB

4. Normalization of the Data

The texts I am going to use for language model training and analysis are live written speech texts, so they may contain information that will be irrelevant to our study, but will also complicate the language model training process. In order to reduce the probability of irrelevant data entering our model and increase its accuracy, the texts should be normalized, i.e., brought to a common form. For this purpose, the texts will undergo several degrees of processing.

4.1 Cleaning the text

Texts taken from an online resource may contain non-written elements, such as residual parts of HTML markup language, which we need to get rid of. For this purpose I used the remove_html_tags(...) function.

4.2 Converting accented symbols

Since our dataset consists of texts in English, it may include words like cafè that include accented accent syllables, so they may not be included when compiling the occurrence matrix if the analysis is to be done considering only the standard English alphabet. To avoid losing some data in the analysis, I normalize such characters to the standard English alphabet using the convert_accented_chars(...) function.

4.3 Removing contractions

The texts of English Internet communication assume the use of colloquial style of speech among others, so it is likely that the texts will often contain contractions of negative particles and auxiliary verbs such as: don't, It's, etc. Failure to take such abbreviations into account when analysing the data can lead to a large statistical error, as these abbreviations may contain, among other things, negative particles that are important for analysing the tone of the text. Therefore, I use expand_contractions(...) function as part of text normalization. I will use the file "contractions.py"[2] as a model for converting contractions to their full form.

4.4 Removing special characters

A standard step in text normalisation is to remove special characters. However, we cannot use a standard function, for example in the nltk library, as it removes all characters by default, including the exclamation mark "!", but in the current language model it is important to consider the occurrence of this character, as it can significantly affect the emotional colouring of the comment we are processing. Thus, the procedure of removing special characters will be formalised through the remove_stopwords(...) function using regular expressions, with a specified list of characters to be removed.

4.5 Stop Words Removal

When removing stop words, we must, as in the previous point, take into account that some words belonging to standard stop word lists, such as negation words, may be important for tone detection, so it is necessary to work with the stop word list before removing stop words from the text. In my research I use the nltk library word list for English, from which I have removed negative particles as well as abbreviations of these particles like don't, isn't, etc.

4.6 Stemming or Lemmatization

To simplify text processing, the method of stemmatisation is also often used, which is based on the fact that the most frequently used affixes are "cut off" from words, which helps to reduce word length and processing time. However, this method can lead to the creation of non-existent words, which in turn can affect the accuracy of the results obtained.

Therefore, in this case, the lemmatisation method is more suitable for us, as it provides not only for bringing words back to their original bases, but also takes into account the context in which the word is located, preserves information about the part of speech, so it more accurately preserves the tones of words, so that on the basis of the analysis of their tones we can make more accurate conclusions about the tone of the text as a whole.

4.7 Text Normalization Pipeline

To simplify the normalisation of the large amount of text that our corpus includes, I will combine all the above functions into one normalisation_pipeline(...) function. This will speed up the automation.

[2] https://github.com/gayathri1462/Suven-Consultants-and-Technology/blob/main/Analysing%20Movie%20Review%20Using%20NLP/contractions.py

5. Research Approach

While searching the literature and research on similar topics, I identified the two most popular approaches that are used in sentiment analysis of text corpora: Lexicon based approach and BERT (Bidirectional Encoder Representations from Transformers) Neural Network model.

Lexicon-based approach in NLP:
A lexicon-based approach in natural language processing (NLP) relies on pre-built dictionaries or lexicons to analyse text. It involves the use of word lists or databases that associate words or phrases with specific sentiment scores, categories, or other linguistic attributes. This approach doesn't rely on machine learning algorithms to understand language but rather uses the knowledge encoded in the lexicon.

For sentiment analysis, for example, a lexicon-based approach might assign a positive or negative sentiment score to each word in a sentence and then aggregate these scores to determine the overall sentiment of the sentence or document. Lexicon-based methods are often interpretable and can work well for certain NLP tasks, but they may struggle with nuanced language and context-dependent meanings.

BERT (Bidirectional Encoder Representations from Transformers) neural network in NLP:
BERT is a state-of-the-art deep learning model in NLP developed by Google. It belongs to the Transformer architecture family. BERT is pre-trained on a massive amount of text data and can understand the context and relationships between words in a bidirectional (both left and right) manner. This bidirectional understanding of language helps BERT capture nuances and context-dependent meanings effectively.

BERT's pre-training involves two main tasks:
1. **Masked Language Model (MLM):** BERT learns to predict missing words in a sentence, which encourages it to understand the relationships between words.
2. **Next Sentence Prediction (NSP):** BERT learns to predict whether one sentence follows another in a given text.

After pre-training, BERT can be fine-tuned for specific NLP tasks such as text classification, question-answering, and named entity recognition. Fine-tuning involves training the model on a smaller dataset specific to the target task.

BERT has achieved remarkable results in various NLP benchmarks and tasks because of its ability to capture context and semantics effectively. It's often used as a powerful tool for transfer learning in NLP, where pre-trained BERT models can be fine-tuned on specific tasks, saving significant training time and resources.[3]

In my work, I am going to use the Lexicon-based approach as it will be easier to train the language model, as I will only have to analyse the positive or negative meanings that are already encoded in the lexical meaning of the word. While BERT neural network is much more complex in training, it collects much more data not only on lexical units but also on

[3] Domadula, P. S. S. V., & Sayyaparaju, S. S. (2023). Sentiment Analysis Of IMDB Movie Reviews: A comparative study of Lexicon-based approach and BERT Neural Network model (Bachelor's thesis). Faculty of Engineering, Blekinge Institute of Technology, 371 79 Karlskrona, Sweden.

syntactic structures, which may increase the accuracy of my research, but it will also reflect in the fact that I will get too much secondary data, which will complicate and slow down the interpretation of the obtained results of training the language model by sentiment analysis of the corpus of texts.

A sample of the necessary code for lexically oriented analysis can be found in the nltk library. In the sentiment_analyzer.py file, which bases its conclusions about the tone of an utterance on the dictionary of positive and negative terms from the vader.py file of the library.

For my project I will use only that part of the code from the library that determines the level of positive or negative colouring of the response, and I will ignore the other models that are necessary to obtain additional information about the character of the utterance, as they are not relevant for the purposes of this project.

Relevant papers use more unique approaches where authors create their own lexical lists to determine the emotional content of the feedback[4], use other ways to normalise the data[5], and conduct multiple tests with different approaches to find the one that best meets their research objectives.

In addition to the method of data collection, it is also important to mention that the way in which the data is handled can vary considerably. The aim of this paper is to automatically sentiment analyse film reviews, but the data obtained from such studies can not only be used to train language models, but also to try to predict what kind of reviews a film will receive depending on, for example, which actors play the lead roles, and with additional data, such as box office receipts, it is possible to infer which themes are popular with viewers in order to use this data when planning new productions. This means that the use of sentiment analysis has not only a broad scientific application, but also possibly even broader commercial applications.[6]

In order to analyse the data more clearly, I decided to visualise the data through a scatterplot. This graph seemed to me the most appropriate for my project, as it not only reflects the amount of data analysed, but by the degree of their scatter and crowding one can draw additional conclusions about the evenness of the distribution of the tone of opinion in the selected data slice.[7]

6. Results Interpretation

[4] Arora, K., Gupta, N., Pathak, S. (2023). Sentiment Analysis on IMDb Movies Review using BERT. In Proceedings of the Fourth International Conference on Electronics and Sustainable Communication Systems (ICESC- 2023) (pp. [page range]). IEEE Xplore Part Number: CFP23V66-ART; ISBN: 979-8-3503-0009-3.

[5] Derbentsev, V. D., Bezkorovainyi, V. S., Matviychuk, A. V., Pomazun, O. M., Hrabariev, A. V., Hostryk, A. M. (2022). A comparative study of deep learning models for sentiment analysis of social media texts. In M3E2-MLPEED 2022: The 10th International Conference on Monitoring, Modeling Management of Emergent Economy . Virtual, Online.

[6] T. Young, D. Hazarika, S. Poria, andE. Cambria, “Recent trends in deep learning based natural language processing[review article],” IEEE Com- put. Intell. Mag., vol. 13, no. 3, pp. 55–75, 2018.

[7] Ivanov, N. (2022, January 6). Comparative Analysis of Comment Senti- ment on YouTube (Caution: Strong Language). Habr [Online]. Available at: https://habr.com/ru/articles/599445/ (Accessed: 16.09.2023).

In the examples of opinion mining research that I have read, the most common method used is to validate the model on pre-processed data and then compare the results to the actual distribution of tone values and then it becomes clear that how accurately the built model is able to determine the tone of the feedback.

First, I applied the developed model to a slice of data from the supervised category. We know from the beginning that of the 25000 reviews that are available in this section of the database, we have exactly half positive and half negative, i.e. 12500 on one side and 12500 on the other.

After applying my model I got the following results:

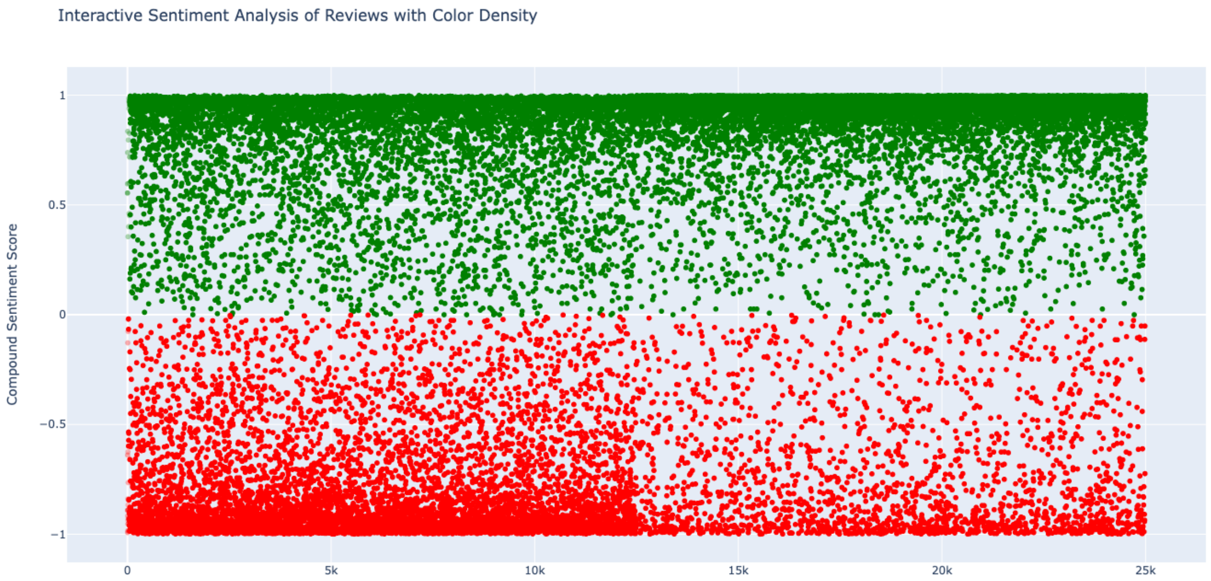

Figure 1 Test data scatter plot (green- positive, red - negative)

From the processing results of this corpus, we can see that the language model I developed considered 16262 reviews as positive and 8738 as negative. Since we know that initially they were manually divided into equal groups of 12500 reviews of both tones, at this stage we can assume that my model tends to identify the tone as positive more often. We'll talk more about this in the error interpretation block.

7. Error Interpretation

If you compare the data I obtained by applying my language model on the test dataset, you will notice that the model identified a much larger number of comments in the positive category than was identified manually (16262 and 12500 respectively). This is most likely due to the lexicon-centric approach I have chosen, as it involves deciding which lexicon we define in advance as positively coloured and which as negatively coloured. Since I did not modify the lexicon dictionary in any way, but simply used the nltk dictionary built into the library, this could also affect the result and make it unpredictable.

This problem can be addressed by adjusting the lexicon list that is considered positively or negatively biased. When dealing with an unpredictable dataset where it's impossible to limit the lexicon list through simple logical enumeration, as is the case in internet communication where people may use a wide range of non-standard vocabulary, slang, figurative language, and even the same words to describe both negative and positive events.

In such cases, to reduce the number of erroneous sentiment analysis results, one can train the model using already evaluated texts. This would automatically compile a lexicon list based on which words had the most occurrences, associated with the ratings given in the reviews containing these most frequent words. Thus, we would narrow down the list of possible vocabulary to the most relevant ones, and also train the model to make conclusions based on vocabulary, the sentiment weight of which has been specifically determined for this type of text, further enhancing the model's accuracy for this specific case.

So, **the first type of error in my project is the use of lexicon created for the automatic processing of a different corpus of texts**.

As for the results of processing the manually unprocessed data subset, considering the known tendency of my model from the test run to classify reviews more often as positive, it can be hypothesized that the distribution of **32096** positive and **17904** negative reviews actually indicates that ehe trend of my model consistently identifying more positive reviews than negative ones in each dataset persists. However, this does not necessarily mean that it's an accurate representation.

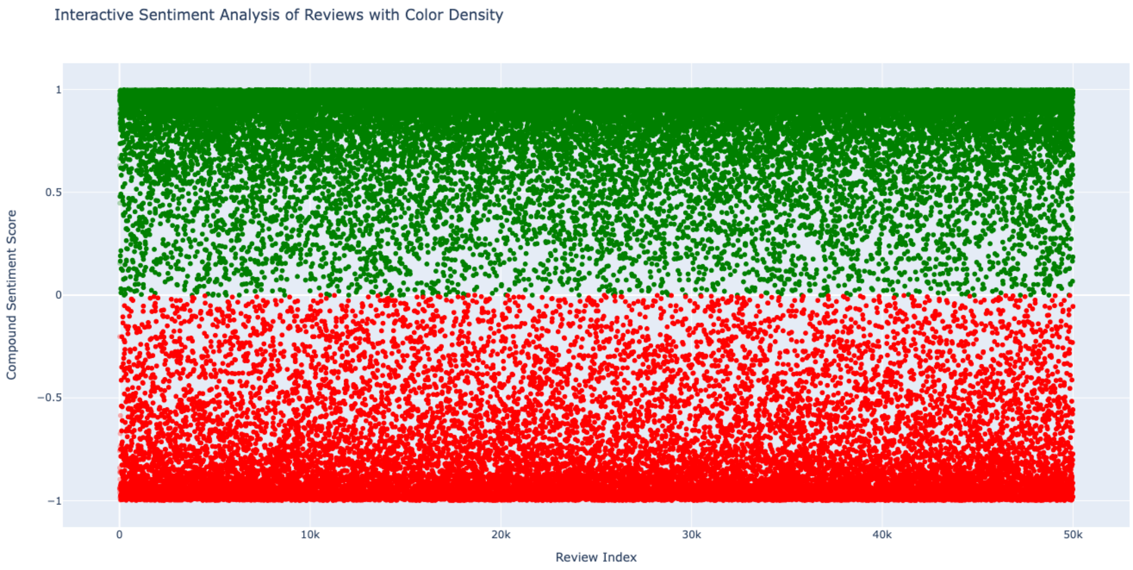

Figure 2 Unsupervised data scatter plot (green- positive, red - negative)

Additionally, we can observe that as the total number of analyzed reviews increases, the gap between the number of positive and negative reviews widens. This might indicate that errors made in the initial stages of the project have a more pronounced effect on the data conclusions when scaled up.

In other words, **the second error lies in the fact that the accuracy of the model's determination cannot be assessed accurately**. With a simplistic lexical sentiment analysis, we do not consider the context of sentiment in relation to the movie rating. This leads to our model making more categorical judgments and excluding scenarios where a user might use "negative" vocabulary while giving a high rating.

8. Conclusion

In conclusion, it can be said that sentiment analysis of various types of texts is a well-researched field in computational linguistics. Thanks to this, there are now numerous convenient tools and approaches that can be easily adapted to specific tasks. The application of such data is also widespread, both in academic and commercial environments.

To achieve the most accurate results, it is recommended to conduct a preliminary linguistic analysis of the specific task at hand. This helps determine which tools are most appropriate and how they can be adapted to further enhance accuracy.

Regarding my project, when reviewing the results I have obtained, it becomes evident that the accuracy of my project could be improved. The tools and approaches I am using are too generic and do not effectively filter out noise in data processing, such as unintended inclusion of irrelevant data during automatic analysis. Unfortunately, at this stage of my exploration into automated data processing systems in text corpora, I lack the knowledge and skills for a more successful adaptation of tools to my project. However, this suggests that in the future, I will be able to use the data I have collected to validate the accuracy of a new language model, should I decide to repeat my research with the same data.