



UNIVERSITÀ DI PISA

DIPARTIMENTO DI INFORMATICA

Human Language Technologies - Laurea Magistrale in Informatica

HLT Project Report

Sentiment Analysis and Topic Modeling on Amazon Reviews

Authors:

Angelo Nardone (548522) - a.nardone5@studenti.unipi.it

Matteo Ziboli (559268) - m.ziboli@studenti.unipi.it

Riccardo Marcaccio (684089) - r.marcaccio@studenti.unipi.it

Abstract

The following report outlines all the crucial steps addressed in the HLT project: from data exploration to the analysis of our NLP tasks results. The project was developed entirely in Python, utilizing both standard Python scripts (.py files) and Jupyter Notebooks (.ipynb) for code implementation and analysis. You can find our code on the following GitHub page: [HLT-Sentiment-Analysis](#) [1]. The main objective of this report is to provide a clear exposition of each step performed, explaining the choices made and illustrating the results obtained. The continuation of the report will then include a description of the datasets used, an explanation of the tasks we aimed to solve, how we solved them and an analysis of the results obtained.

1 Introduction

The analysis of an emotional reaction stimulated by an event has increasingly become a subject of research, not only in traditional disciplines such as psychology, but also in the field of **Natural Language Processing (NLP)** with machine learning algorithms [2]. In this context, **sentiment analysis** is a type of text classification that aims to categorize pieces of text into one of the following classes: positive, negative, or neutral. The categorization can involve different textual levels, ranging from portions of text of document size to that of single words. Clearly, a sentence in its entirety can receive a positive or negative label based on the words contained within it. At the same time, one can isolate specific aspects of the document being evaluated, which manifest in certain words or phrases, to conduct a more specific analysis. For instance, the sentence "*this ice cream is delicious*" contains the adjective "*delicious*", which conveys the speaker's opinion and can be used to label the entire sentence as positive. Conversely, if one wants to evaluate, for example, the service of a restaurant, the classification should be narrowed to those pieces of text that specifically refer to the service, rather than the food, the location, or other services in general [3]. To achieve this, various **machine learning algorithms** are used today, which can be divided into three broad categories: supervised learning, semi-supervised learning, and unsupervised learning. In each case, these algorithms are capable of handling vast amounts of data and extracting insights that would be otherwise impossible to analyze manually.

Another type of analysis that can be combined with sentiment analysis is **topic modeling**. This machine learning technique is capable of identifying the topics present in a set of documents through unsupervised learning. In the context of NLP, which we are addressing, topic modeling has been used for tasks such as text summarization and sentiment analysis [4]. Generally, its use can **reveal trends in documents**, which, for example, in an industrial context, helps trace the evolution of user opinions and preferences regarding products. Essentially, companies can benefit from this by enhancing their sales strategies.

So, during this project, we explored the techniques cited above on a real dataset to demonstrate their functionality and potential, as well as for educational purposes. We focused on sentiment analysis as the primary task and topic modeling as the secondary task. These tasks were applied to a set of **Amazon reviews** to showcase how the application of machine learning techniques can significantly enhance the company-consumer interface. For the company, this enhancement comes from direct insights into what customers like and dislike, and, where possible, the reasons behind these opinions. For the consumer, the improvement arises from the feedback provided by other users, which can guide their purchasing decisions based on the quality of the products they are seeking. Given the characteristics of our dataset (see Section 2), the type of classification considered in this research is **binary**, aiming to label the available data as either positive or negative.

Related Works

Sentiment analysis has always garnered interest from its earliest applications [5] to the present day [6]. The reasons for this interest can be traced both to the understanding that building an intelligent machine capable of mimicking human behavior requires the ability to recognize **human emotions**, and to the importance of practical applications such as user-oriented online merchandise sales, a phenomenon of recent decades (for a review in the literature [7]). Various models have been applied in sentiment analysis. For example, Pang et al. [8] classified movie reviews using algorithms such as the **Naive Bayes classifier** and **Support Vector Machine (SVM)**. In other cases, more complex architectures like **Recurrent Neural Networks**, particularly the **Long Short-Term Memory (LSTM)** variant, have been employed [9].

Recently, **Large Language Models (LLMs)** have been used for this type of task in zero-shot or few-shot learning contexts [7]. However, the use of such models has been relatively limited. **BERT**, an important LLM introduced in 2018 [10], has been used occasionally for sentiment analysis tasks (e.g., [11], [12]). It has two variants that differ in the number of layers (transformer blocks), hidden layers, self-attention heads, and total trainable parameters (**BERTBASE** e **BERTLARGE**) [10]. In [12], BERT, both as a pretrained and fine-tuned model for a fine-grained sentiment classification task on the **Stanford Sentiment Treebank (SST)**, outperformed architectures such as convolutional and recurrent networks. This success is attributed to its transfer learning ability, enabling rapid reuse on datasets linguistically different from those on which it was pretrained. In particular, this capability will also be analysed in this report, as we used BERTBASE as the main part for our sentiment analysis model. In particular, we used a pre-trained BERT, to which we added some feedforward layers, in order to take a sentence as input and return the desired binary result.

As mentioned in the introduction, topic modeling has also been applied in text analysis and it is done so through various models that differ based on many factors, including the type of corpus and the topics identified. Generally, there are four models that perform this type of task: **algebraic**, **fuzzy**, **probabilistic**, and **neural** [4]. No single model consistently outperforms the others; rather, each is used in different settings that require distinct approaches (for a useful overview, see [4]). One type of probabilistic model is **Latent Dirichlet Allocation (LDA)**, which is utilized in this project. It identifies latent topics present in each document, with each topic having a multinomial distribution over the vocabulary of words [13]. Then, as a generative model, LDA generates a document one word at a time, by sampling a topic from the topics previously identified.

Structure of the Report

In Section 2, we will introduce the dataset used to develop our work. In Section 3, we will present the model architectures employed for sentiment analysis and topic modeling tasks. In Section 4, we will detail the experiments performed. In Section 5, we will analyze the results obtained, comparing them with the state-of-the-art where possible. Finally, in Section 6, we will provide our conclusions and possible future considerations.

2 Data

The dataset used is a large dataset available on [Kaggle](#) [14]. The dataset was composed by taking several amazon reviews, and in particular the information used to build the dataset were:

Column	Description
<code>title</code>	It contains the titles of the reviews.
<code>text</code>	It contains the entire text of each review.
<code>polarity</code>	It contains 1 for the negative review, or 2 for the positive ones.

The polarity was constructed based on Amazon's five-score rating system, with scores 1 and 2 aggregated as negative and scores 4 and 5 as positive; the score 3 was dismissed because of the binary labeling goal of the dataset. So, the former were classified by the authors of the dataset as 1 (negative), while the latter were classified as 2 (positive). The dataset consisted of 1,800,000 data points in total.

Before feeding our data into the models for our NLP tasks, we performed several preprocessing and exploratory steps. Firstly, we **cleaned the textual data** by removing elements that could interfere with the models. This included eliminating links, email addresses, and special characters from the text. We also checked for duplicate rows and rows with null values. No duplicates were present, while we found a total of 207 null rows which we eliminated.

Next, we explored the dataset in various ways. We ensured that the data in the polarity column, used as the discriminant in our sentiment analysis, was **balanced**, as shown in the Figure 1(c). We analyzed the **most frequently used words** in our dataset, both in the review titles and the full reviews, using the [WordCloud](#) [15] library. We examined the results by dividing the texts into negative and positive reviews, with some examples shown in the Figure 2. Additionally, we studied the **distribution of document lengths** in the titles and full reviews, investigating whether these distributions correlated with the sentiment (positive or negative) of the reviews. The final conclusion was that the length distribution of the reviews did not depend

on their sentiment, as is shown in Figure 1(a) and 1(b).

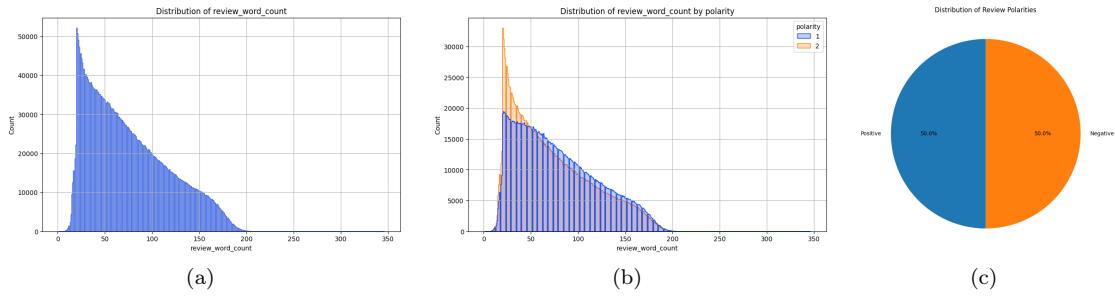


Figure 1: (a) Chart showing the distribution of review text lengths. (b) Chart displaying the same distributions, separating positive and negative reviews. (c) Distribution of values in the polarity column.

As a final step, we **reduced the size** of our dataset by keeping only 200,000 rows, taken randomly so that the polarity values remained balanced.. This reduction was necessary due to limited training time on the servers, allowing us to train the models within a reasonable timeframe while still using a substantial amount of data to obtain meaningful results. The cleaned dataset was then saved into a new CSV file, ready for use in our tasks.



Figure 2: (a) Wordcloud results for titles of negative reviews. (b) Wordcloud results for titles of positive reviews.

3 Models

In this section, we will explore the architectures of the models used to complete the tasks of sentiment analysis and topic modeling.

Sentiment Analysis

In the case of sentiment analysis, we will work with the **supervised learning paradigm**. In particular, as mentioned earlier, the classifier we used to perform the sentiment analysis uses a **Large Language Model** (LLM) to preprocess the data. Specifically, we utilized BERT-base-cased in combination with a **three-layer feedforward network** to process our text input and return a probability indicating whether the text represents a negative or positive review. The main libraries used to build our model were [PyTorch](#) and [Hugging Face's Transformers](#). The Figure 3 schematically shows all the components of our model, which we will now analyze from left to right.

The first element of our model is the input text. As presented in the Section 2, the text consists of Amazon reviews and their labels indicating whether the reviews were positive or negative. For the sentiment analysis task, we **used only the titles** of each review, ignoring the full text. This choice was driven by two reasons: to reduce the training time, given our limited server resources, and to demonstrate the power of our model to accurately classify reviews with limited information (clearly, classifying the full review text is simpler than classifying just the title).

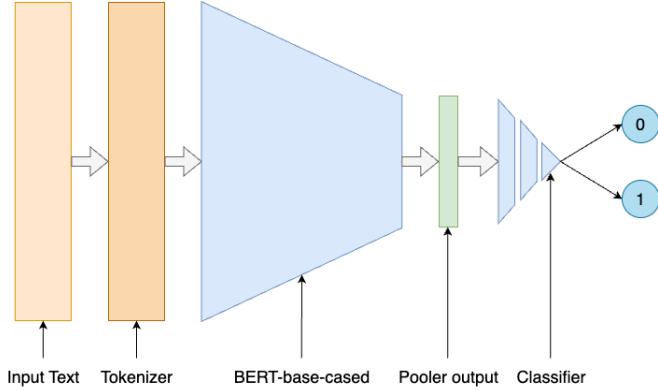


Figure 3: Structure of the model used for the Sentiment Analysis task.

Before being passed to BERT, the titles needed to undergo another preprocessing step, as we had to **tokenize** them. We used the `BERT-base-cased` tokenizer from the `Transformers` library for this purpose. Once tokenized, each title was passed through BERT-base-cased, a LLM composed of 12 transformer encoder layers. The result obtained after this phase was a 768-dimensional **embedding vector** that encapsulated the meaning of the entire sentence. In particular, we achieved this by taking the **pooler output** of BERT.

The final part of our model consists of a standard three-layer feedforward network. The structure of this feedforward network is presented in Table 1. Notice how we used both **dropout**, set equal to 0.1, to prevent overfitting and **batch normalization** [16] to stabilize the learning process. Additionally, the last activation function used is a **sigmoid function**, which ensures that the output is a number between 0 and 1, representing a probability. Our model classified reviews as negative if they had a probability less than or equal to 0.5 and as positive if they had a probability greater than 0.5.

	Input Size	Output Size	Activation Function	Additional Operations
First Layer	768	300	ReLU	BatchNorm1d, Dropout
Second Layer	300	100	ReLU	BatchNorm1d, Dropout
Output Layer	100	1	Sigmoid	-

Table 1: Structure of the feedforward network used for sentiment analysis

Topic Modeling

In contrast to the supervised approach seen earlier, for the task of topic modeling, we adopted an **unsupervised approach**. Indeed, as labels were unavailable to us, we could not rely on them to guide our analysis. Therefore, to achieve our objective of automatically extracting the main themes from the review texts in an unsupervised manner, we employed the **Latent Dirichlet Allocation (LDA)** model.

It's worth emphasizing that, in this case, we worked with the **entire text of the reviews** rather than just the titles. This decision was driven by the need for a greater quantity of information to effectively extract topics from the data.

Our model comprised three main parts. In the first part, we conducted **additional preprocessing of the texts** beyond what was done in Section 3. This preprocessing was crucial to optimize the results of the model. We removed punctuation marks and tokenized the sentences into words using the `Gensim simple_preprocess` function. Additionally, we built **bigram** and **trigram** models to capture more complex semantic relationships within the texts. These models were used to enhance the understanding of context and semantic associations between words. Finally, we **removed stopwords** and **performed lemmatization**. Stopwords were removed to focus attention on the most meaningful words. Lemmatization was carried out to reduce words to their base forms, retaining only the parts of speech relevant for the analysis of themes: specifically, nouns, adjectives, verbs, and adverbs.

In the second part, utilizing the `corpora.Dictionary` function from `Gensim`, we constructed a **dictionary of terms** and a corpus of documents to represent the data in a format compatible with the LDA model. In the last part, we defined our LDA model, again utilizing `LdaModel` function from `Gensim`. This model

could take input parameters and return a variable number of topics based on our specifications. Using a probabilistic distribution-based method, the LDA model assigned each document a probability distribution over the topics, enabling the identification of the primary themes addressed in the reviews.

4 Experiments

In this section, we will present the experiments conducted for both models, along with the parameters explored and their values. Additionally, we will discuss the metrics used to determine the best parameters for our model.

Sentiment Analysis

For the sentiment analysis task, we divided our dataset into three parts: **training set**, **validation set**, and **test set**. Specifically, we used 20% of the total data for the test set, and from the remaining 80%, we allocated 10% for the validation set. We ensured that the splits were stratified to maintain balanced class distribution in the polarity column, which is our target variable. The test set was used to evaluate the final results, which we will discuss in the Section 5. The training set was used to train the model, and the validation set was used to evaluate the performance during training.

During the training phase, we primarily used **accuracy** as the metric to evaluate and select the best parameters for our model. This choice was driven by our goal to achieve the highest classification performance, and given that our dataset was balanced, accuracy was the most appropriate metric. We also monitored the validation **loss**, as well as the training loss and training accuracy, to ensure comprehensive evaluation.

For training the model, we used **Binary Cross-Entropy (BCE)** loss. This loss function is widely used in binary classification tasks in NLP for several reasons: it is differentiable and handles probabilities effectively. In addition, we employed the **Adam optimizer**, which provided efficient computation and adaptive learning rates.

Our training process was divided into two main phases. In the first phase, we performed **fine-tuning**, adjusting both the parameters of the feedforward classifier and the BERT encoder. This allowed the pooler output from BERT, which in any case captured the meaning of the sentences well from the start, to better capture the specific context of our dataset. In the second phase, we **froze the BERT parameters** and trained only the feedforward classifier. This approach helped reduce training time, as BERT has significantly more parameters than the feedforward network, and it also helped prevent overfitting.

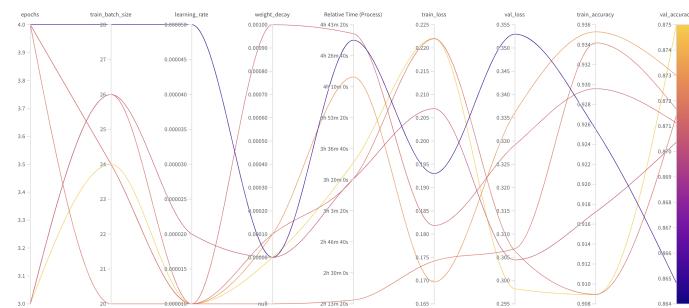


Figure 4: Figure created using Weights & Biases (wandb) that illustrates the variations in results across the top 7 training runs based on different hyperparameter settings.

Finally, we explored various **hyperparameters** during training, including batch size, number of epochs, learning rate, and weight decay. We used the [Weights & Biases](#) library to log the results of each training run, storing them in a centralized repository for quick analysis. Figure 4 and Table 2 show the values obtained from the training runs that yielded the best results. Ultimately, we selected the configuration that provided the best performance, achieving an accuracy of 0.875 on the validation set.

Run Name	Runtime	Epochs	Learning Rate	Batch Size	Weight Decay	Val Accuracy
fresh-fire-10	3h 26m 10s	3	0.00001	26	0.0001	0.8714
faithful-jazz-9	3h 26m 20s	3	0.00002	26	0	0.8704
dazzling-galaxy-8	4h 40m 58s	4	0.00005	28	0	0.8648
fiery-disco-4	3h 35m 11s	3	0.00001	24	0	0.875
winter-glitter-3	4h 44m 16s	4	0.00001	0.001	0.1818	0.8711
whole-eon-2	4h 21m 4s	4	0.00001	24	0.0001	0.873
unique-flower-1	2h 19m	4	0.00001	20	0	0.8717

Table 2: Results of the top 7 training runs with various hyperparameters.

Topic Modeling

The approach to evaluating the best parameters for our topic modeling task differed significantly from that of sentiment analysis task. In fact, since our task was unsupervised, we couldn't define a loss function as in supervised scenarios. Instead, we relied on two popular metrics: **perplexity** and **coherence score**.

Perplexity measures how well the model predicts a sample, with lower values indicating better performance. Coherence score, on the other hand, assesses the interpretability of topics by measuring the degree of semantic similarity between high-scoring words in the topic. Given the qualitative nature of our analysis and the tendency for highly imbalanced topics when maximizing perplexity, we prioritized coherence score for our evaluations.

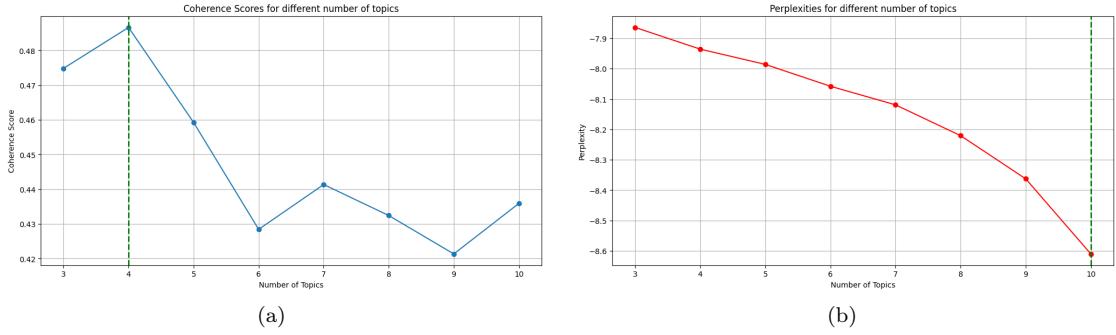


Figure 5: (a) Coherence score value as a function of the number of topics. (b) Perplexity value as a function of the number of topics.

In constructing the LDA model, we explored several parameters, primarily focusing on the number of topics. Through a mini-grid search ranging from 3 to 10 topics, we determined that 4 topics yielded the highest coherence score, optimizing the interpretability of the resulting topics. Additionally, we experimented with other parameters such as `chunksize`, defining the size of data blocks processed in a single pass (set to 100), and `passes`, the number of passes the model makes over the entire corpus during training (set to 10). We also set the `alpha` parameter to 'auto' to allow the model to learn an optimal value for alpha, controlling the distribution of topics per document. Furthermore, `update_every` was set to 1, ensuring the model updates after each pass through the corpus. Finally, we enabled `per_word_topics` equal to true, to provide the probability distribution of each word for each topic, enhancing the granularity of the model's output.

5 Results

In this section, we will evaluate the results obtained with both tasks, and where possible, compare them with the state of the art.

Sentiment Analysis

In this section, we finally evaluate our model by comparing it with other models. To evaluate our model, we used the results obtained on the test set with the parameters that yielded the best results on the validation set. Once again, we used **accuracy** as the primary metric since, with a dataset featuring balanced classes, it represents the most suitable metric for us. However, we also assess other metrics such as **precision**,

recall, and **F1 score**. To evaluate our model, we utilized methods from the `sklearn` library.

We considered several models for comparison. Three of these models were defined by us (they can be found on our GitHub [1]), mainly to evaluate our main model, but also for teaching purposes. These are: a **lexicon-based model**, in particular using **Vader** [17], a model using **Multinomial Naive Bayes** and one using a simple **multilayer perceptron**. Additionally, we obtained results from other models available from the same Kaggle link where we acquired our dataset. Specifically, we compared our results with results from a model based on **RoBERTa** [18], and results from **decision tree**, **random forest**, and **SVM models** [19]. It should be noted that our model outperformed all models that performed the task using only review titles, as we did. Moreover, it achieved results better than, and in only one case equal to, those of models that also used the full texts of the reviews for classification. We consider these results extremely positive, as we either improved upon the results of other models or achieved similar performance while using significantly less data and information.

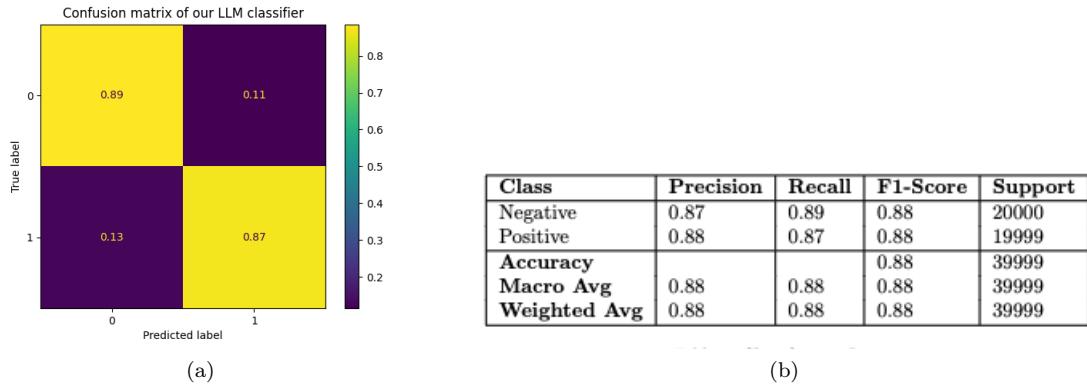


Figure 6: (a) Normalized confusion matrix for the results of our model. (b) Table reporting the results of our model on various relevant metrics.

The results of our model are shown in Figure 6 using a **confusion matrix** and a **classification report**. Instead, comparisons with other models are presented in Table 3.

Model	Accuracy	Precision	Recall	F1 Score	Data Used
Our Model (bert-base-cased)	0.88	0.88	0.88	0.88	Title only
Multinomial Naive Bayes	0.82	0.82	0.82	0.82	Title only
Lexicon (Vader)	0.67	0.74	0.67	0.64	Title only
Multilayer Perceptron	0.82	0.83	0.82	0.82	Title only
RoBERTa	0.84				Title only
RoBERTa	0.88				Title+Text
Decision Tree	0.76	0.76	0.76	0.76	Title+Text
Random Forest	0.86	0.86	0.86	0.86	Title+Text
SVM	0.88	0.88	0.88	0.88	Title+Text

Table 3: Evaluation results of different models.

Topic Modeling

Finally, we present the results obtained from our LDA model. Given the nature of the task, we were unable to compare our model with others, and we did not have access to metrics on which to draw objective evaluations of our model. What we did, as is usually done, was to analyze the contents of the topics identified by our model and study how these topics were distributed and influenced our data. As seen in the Section 4, the ideal number of topics to find was 4, so we set the model to return exactly 4 topics.

First, we tried to understand from the most frequent words of each topic what the main topic was about. The 4 topics found, as shown in Figure 7(a), are: **Product Related Issues**, **Book and Movie**, **Personal Experiences and Opinions**, and **Game and Music**. While the first topic appears to be the most **objective**, concerning physical product issues or order delivery, the other three topics seemed to deal with more **personal issues**. In particular, the topics on Book and Movie and Game and Music deal with specific

issues related to these categories, probably indicating that the product does not align with the consumer's tastes. The Personal Experience topic, on the other hand, seems to deal with various issues between product and consumer, but they remain within the sphere of personal problems such as: product quality-price ratio, product different from expected, etc.

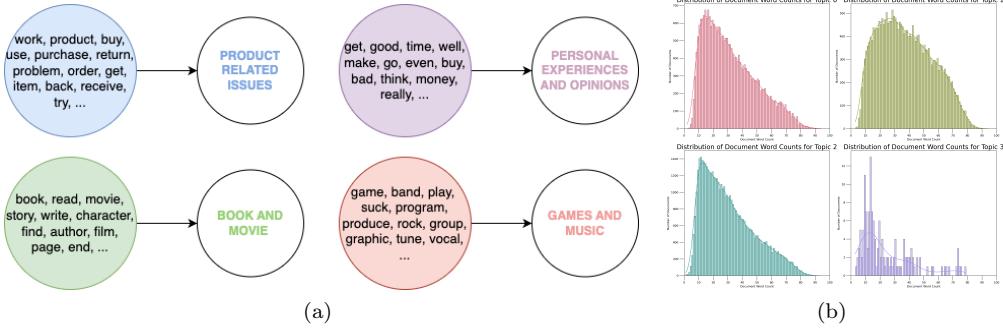


Figure 7: (a) Most frequent words of each topic and main topic. (b) Distribution of review lengths for each topic.

After studying the topics of the 4 topics, we went on to analyze their distribution within the dataset. We saw, for each review, what percentage of words was associated with the dominant topic. We also saw the distribution of the **lengths of the reviews** for each topic. In this case, as shown in the Figure 7(b), it can be seen how, in addition to having fewer elements, the distribution of the length of the reviews in the last topic is also the least regular. We then used special prints to better visualize some aspects already seen: we again used **Word Clouds** to print the most frequent words for each topic, and we did the same print using colored histograms; or we reprinted the first 10 reviews, coloring the words of each review with the color of the topic to which it belonged. This last print can also be seen in Figure 8(a). Finally, we analyzed how the **topic clusters** were distributed in 2D using **t-SNE** function. The result can be seen in Figure 8(b).

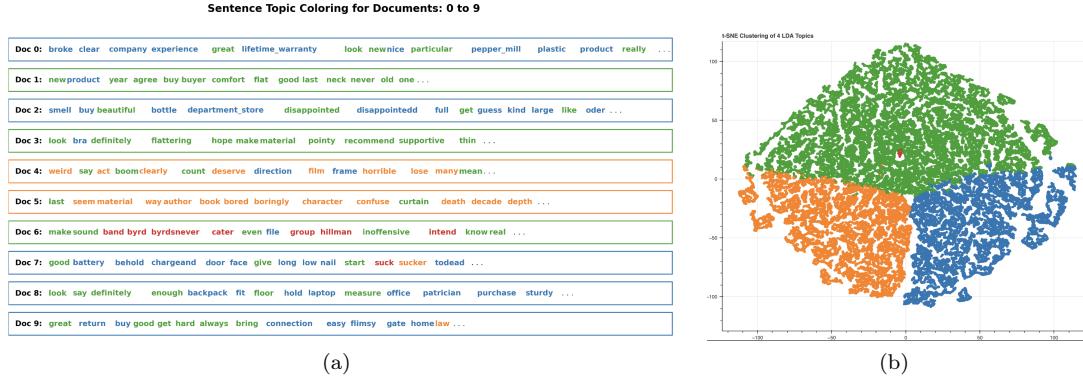


Figure 8: (a) First 10 reviews, coloring words to associate them with the corresponding topic. (b) 2D visualization of topic clustering.

6 Conclusion

In this report, we illustrated our work in analyzing text data sourced from Amazon reviews. Specifically, we have demonstrated the effectiveness of employing advanced **Natural Language Processing (NLP)** approaches to explore and understand customer reviews on platforms like Amazon. Through the utilization of a language model such as **BERT** (specifically, `bert-base-cased`) we have achieved remarkable results even with limited input information, focusing only on review titles and a subset of the data to achieve approximately 90% accuracy. Additionally, we employed the **Latent Dirichlet Allocation (LDA)** model to delve into the primary topics within negative reviews.

These approaches, as we have already written, can be helpful for both companies and consumers. For companies, they offer insights to enhance product quality based on customer feedback, while for consumers, they provide valuable insights prior to making purchasing decisions.

However, it is clear that these methodologies can be improved. For example, due to the computational constraints inherent in this educational project, we were limited in our ability to fully explore different hyperparameter configurations. Furthermore, our classifier can be improved through various techniques: such as performing a **grid search**, using **more data** as input, or considering the **totality of reviews** rather than just their titles. Similarly, in the realm of topic modeling, access to more data could yield even more insightful results. Moreover, exploring **other approaches** to address this task could present an intriguing challenge for future endeavors.

References

- [1] Angelido. "HLT-Sentiment-Analysis: Sentiment Analysis project repository." In *GitHub*. [GitHub Repository](#). (2024)
- [2] S. Lai, X. Hu, H. Xu, Z. Ren and Z. Liu. "Multimodal sentiment analysis: A survey." In *Displays, Volume 80, pag. 1-15*. (2023)
- [3] S. Kumar, P. P. Roy, D. P. Dogra and B. G. Kim. "A Comprehensive Review on Sentiment Analysis: Tasks, Approaches and Applications." In *arXiv preprint arXiv:2311.11250*. (2023)
- [4] A. Aly, G. Eman, M. Walaa and H. Ahmed. "Topic modeling algorithms and applications: A survey." In *Information Systems, vol. 112*. (2023)
- [5] H. Yu and V. Hatzivassiloglou. "Towards Answering Opinion Questions: Separating Facts from Opinions and Identifying the Polarity of Opinion Sentences." In *Conference on Empirical Methods in Natural Language Processing*. (2003)
- [6] A. Yadav and D. K. Vishwakarma. "Sentiment analysis using deep learning architectures: a review." In *Artificial Intelligence Review, 53: 4335-4385*. (2019)
- [7] W. Zhang, Y. Deng, B. Liu, S. J. Pan and L. Bing. "Sentiment Analysis in the Era of Large Language Models: A Reality Check." In *arXiv:2305.15005*. (2023)
- [8] B. Pang, L. Lee, and S. Vaithyanathan. "Thumbs up? Sentiment classification using machine learning techniques." In *Conference on Empirical Methods in Natural Language Processing: 79-86*. (2002)
- [9] W. Tan, W. Xinyu and X. Xinyu. "Sentiment analysis for Amazon reviews." In *International Conference*. (2018).
- [10] J. Devlin, M. Chang, K. Lee, and K. Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In *North American Chapter of the Association for Computational Linguistics*. (2019)
- [11] M. Munikar, S. Shakya and A. Shrestha. "Fine-grained Sentiment Classification using BERT." In *Artificial Intelligence for Transforming Business and Society (AITB), Kathmandu, Nepal: 1-5*. (2019)
- [12] F. D. Souza and J. B. d. O. e. S. Filho. "BERT for Sentiment Analysis: Pre-trained and Fine-Tuned Alternatives." In *V. Pinheiro et al. Computational Processing of the Portuguese Language. Lecture Notes in Computer Science, vol 13208*. (2022)
- [13] R. Alghamdi and K. Alfalqi. "A Survey of Topic Modeling in Text Mining." In *International Journal of Advanced Computer Science and Applications, vol. 6*. (2015)
- [14] K. Jain. "Amazon Reviews." In *Kaggle*. [Online Dataset](#). (2024)
- [15] Andreas Mueller. "Word Cloud: A Command Line Interface for Creating Word Clouds." *Journal of Open Source Software 3.26: 781*. [WordCloud Command Line Interface](#). (2018)
- [16] S. Ioffe and C. Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In *arXiv preprint arXiv:1502.03167*. (2015)
- [17] V. Bonta, N. Kumaresan and N. Janardhan. "A comprehensive study on lexicon based approaches for sentiment analysis." In *Asian Journal of Computer Science and Technology 8.S2: 1-6*. (2019)
- [18] Famalouiz. "Sentiment Analysis Project." *Kaggle*, Available at: [here](#). (2024)
- [19] Shivangamsoni. "Sentiment Analysis TF-IDF." *Kaggle*, Available at: [here](#). (2024)