University *of Ljubljana*
Faculty *of Computer and Information Science*

Qualitative Research on Discussions - text categorization

Sanil Safić, Miha Šircelj, Jan Topolovec

Abstract

This report presents an advanced method for analyzing online discussions of ”The Lady, or the Tiger?” using Large Language Models (LLMs) for text categorization. Our methodology includes dataset coding, model refinement on high-performance computing (HPC) infrastructure, and iterative performance enhancement compared to human coders. Additionally, we leverage fine-tuned LLMs to provide explanations for the identified categories, improving interpretability. This research aims to develop robust models for accurately categorizing discussions, thereby enhancing our understanding of discourse dynamics and contributing to the broader field of qualitative discourse analysis.

Keywords

Text categorization, LLMs, Natural language processing

Advisors: Slavko Žitnik

Introduction

The objective of this project is to utilize Large Language Models (LLMs) for the qualitative analysis of discourse, with a focus on text categorization. Our approach involves a comprehensive review of relevant literature, thorough data exploration, meticulous model fine-tuning, and rigorous performance evaluation. By employing these methods, we aim to create highly reliable models that not only perform well on the provided dataset but also generalize effectively to other online discussions. Our process includes iterative comparisons with human-annotated labels to ensure the models' accuracy and reliability at each step. This study contributes to the field of Natural Language Processing (NLP) by providing insights and methodologies for the effective categorization of textual data in online discussions.

Related work

Numerous studies have delved into research methods for analyzing discussions and categorizing text. Researchers have explored different approaches like thematic analysis and machine learning algorithms to better understand how discussions evolve and how text can be categorized effectively.

In this section, we will discuss three related articles. The first article concentrates on supervised and traditional term weighting methods for automatic text categorization, while the other two articles focus on the utilization of GPT engines for text analysis and categorization.

The first article [1] investigates various term weighting methods to enhance automatic text categorization performance. It evaluates both supervised and traditional (unsupervised) methods on benchmark datasets like Reuters-21578 and 20 Newsgroups. Introducing a new supervised term weighting method called *tf.rf*, which combines term frequency with relevance frequency, the study aims to improve term discriminating power by considering the distribution of relevant documents in the collection. Results indicate that while traditional methods like *tf.idf* showed mixed performance, the proposed *tf.rf* method consistently outperformed others across different datasets and algorithms, suggesting its effectiveness in enhancing text categorization by better capturing term importance based on document distribution in categories.

The second article [2] introduces TopicGPT, a prompt-based framework that uses LLMs to find topics in a collection of documents. They address the limitation of using bag-of-words topics in other topic modeling methods. TopicGPT operates in two main stages: topic generation and topic assignment. The first one iteratively uses the LLM to generate descriptive topics based on the dataset, refining and merging them for coherence. In the second stage, the LLM assigns topics to documents using the generated topic list, ensuring accuracy through verifiable quotes and self-correction for errors. Results show that TopicGPT aligns substantially better with human-annotated labels than baselines, LDA and BERTopic. The dataset used for this study includes Wikipedia articles and Congressional bills.

The article [3] undertakes an analysis of Twitter posts to identify some concerns about using Generative AI models, particularly ChatGPT, in education. The authors collected data by finding mentions of ChatGPT using Twitter API - they collected a total of 247,484 tweets. For sentiment analysis, they used fine-tuned RoBERTa model which showed better performance than SVM-based models. The RoBERTa model categorized tweets as positive, neutral and negative, which where subjected to further analysis. They employed BERTopic tool to cluster negative tweets into 200 topics to better expose concerns. The sentiment analysis shows that the majority of users expressed positive or neutral attitude towards ChatGPT in education. Among those expressing reservations about its use, there were some notable topics, such as: Academic integrity, impact on learning outcomes and workforce challenges, specifically expressed by individuals from education and tech.

Data Exploration

In this project, where we aim to create a model for text categorization, we will be referencing the data presented in the form of discussions based on the story "The Lady, or the Tiger?". The author of story does not define its ending consequently leaves room for many interpretations by readers.

The given data includes discussions that are already appropriately categorized according to category definitions. Each comment is initially categorized according to their type - such as "Seminar" (discussion on story interpretation) or "Social" (discussion between users that is not related to the interpretation). Some of these questions also leave room for further interpretation. In the case of questions, they are generally divided into open and close-ended types, while certain discussions can also be labeled as explanations or agreement of previous arguments.

The columns "Message" and "R2DiscussionType" are crucial for our model. The "Message" column contains the actual text of the message exchange in the discussion, ranging from greetings and assumptions to interpretations and feedback. The "R2DiscussionType" column categorizes the type of discussion to which each message belongs. The "R2Uptake" column could also be relevant as it indicates whether a message has been responded to, and if so, how.

In addition to these, we also analyzed the types of discussions within these exchanges. The "R2DiscussionType" column includes categories such as "Seminar", "Deliberation", "Social", "Procedure", "UX", and "Imaginative Entry". The majority of the discussions are categorized as "Seminar" (333) or "Deliberation" (85), so these categories should not pose a problem for training the model. The most challenging category is "Imaginative Entry", with only 18 instances. All other categories have at least around 50 instances or more. All of the categories and their counts can be seen in Table 1.

After a closer look into "R2DiscussionType" values, we notice that some messages have two different values for this categorization. We find five such examples, of which one contains "Social" and "Procedure", one contains "Social" and "Deliberation", and the other four contain "Seminar" and "Deliberation". One of the options is to remove only one of the two categories, but which one? The other option could be to create a new category for each of the two combinations; however, this approach might not work due to the limited number of rows with these combinations. Since our model will predict only one category, the best approach is to remove these five instances. There were also other problems like one time "imaginative entry" was written with a large e (which we fix) and another two where only the first word was present (we deleted these two). We also found out that there is a category "Other" with six entries, which we will keep.

R2DiscussionType	Count
Seminar	333
Deliberation	85
Social	69
UX	47
Procedure	46
Imaginative entry	18
Other	6
Seminar, Deliberation	2
Imaginative	2
Social, Deliberation	1
Deliberation, Seminar	1
Social, Procedure	1
Imaginative Entry	1

Table 1. R2DiscussionType counts

Let's consider the "R2Uptake" column as it might be useful for our model. The data reveals various types including "Affirm", "Elaborate", "Disagree", "Filler", "Prompt", and "Clarify". Most of the messages are labeled as "Affirm" (70%). The "Elaborate" category also has a significant number of instances (28). The other categories have only 4 or 7 instances each. Thus, we can conclude that "Affirm" and "Elaborate" are the most common types of responses with labels. However, most of the responses have no label. So we can conclude that the "R2Uptake" column is not very useful for our model to predict the "R2DiscussionType" values.

Proposed implementation

The first step involves thoroughly exploring the provided dataset of coded discussions. This includes examining the dataset's structure, studying the distribution of categories, and understanding the details of the coded discussions.

After exploring the data, we will start building and refining Large Language Models (LLMs) for text categorization. To handle the computational demands of training large-scale models, we'll use a HPC infrastructure, ensuring efficient model development.

Performance evaluation will be a key component of our

implementation strategy. Our approach will include iteratively comparing and revising our model's performance against that of human coders (categorizers).

We will use a separately fine-tuned LLM to generate explanations for the identified categories. These explanations will undergo qualitative assessment to ensure clarity and trustworthiness, improving the interpretability of our models' predictions.

TF-IDF method

To get things started and a feel for some base results we started off by utilizing the more traditional tf-idf method (source [4]).

Initially, we read the prepared data from the classroom (The Lady Or The Tiger) and preprocessed it - tokenization, converting to lowercase, removing stopwords, etc.

Then we used a class TfidfVectorizer to convert the data to TF-IDF vectors.

Next, we split the data into training, validation and testing sets using a 60-20-20 split. We can see the detailed category split in the table 2.

Category	Train Data	Validation Data	Test Data
Seminar	199	67	67
Social	41	14	14
Deliberation	51	17	17
UX	28	10	9
Procedure	28	9	9
Imaginative entry	10	3	4
Other	4	1	1

Table 2. Category counts for train, validation, and test data

After the data was prepared and split we found the optimal hyperparameters using the grid search method. We utilized GridSearchCV class for this part. We used the C parameter which is the inverse of regularization strength. Tor C we tested the values 0.1, 1, 10 and 100. We also used penalty parameter with l2 value - Ridge and a liblinear solver. Once we got our best performing parameters we used them to train the logistic regression classifier on a training set.

The final step in our tf-idf method for classification was to evaluate the model on validation and test set. We predicted the categories with the model and then generated a classification report using the classification_report method from the sklearn.metrics toolbox. The results of the evaluations are presented in the table 3.

Dataset	Accuracy	Precision	Recall	F1 Score
Validation	0.6612	0.6622	0.6612	0.6111
Testing	0.6033	0.5682	0.6033	0.5803

Table 3. Results of the validation set and test set evaluations

Bert model

The next version of our project was implemented using a pre-trained BERT model. Again, we read the prepared data and did some preprocessing. We then selected a pre-trained model suitable for classification. Next, we divided the acquired data into training, validation and testing sets using a 60-20-20 split. Also, we needed to convert labels to integers (tensors) to be able to use them in the model. The returned value from the model will be a tensor, which will be converted back to the label.

Since we split the data randomly into training, validation, and testing sets, we had to ensure that the distribution of categories in each set was similar. Almost all categories of "R2DiscussionType" were evenly distributed in all three sets (Table 4). Only "Imaginative entry" and "Other" categories were underrepresented in the training set. We decided to keep the distribution as it is since the number of "Imaginative entry" is very low in the dataset. Also, there is a good representation in the validation dataset, so we can see how the model performs in these categories.

Category	Train Data	Validation Data	Test Data
Seminar	191	66	76
Social	48	13	8
Deliberation	47	21	17
UX	31	8	8
Procedure	29	6	11
Imaginative entry	12	4	1
Other	3	3	0

Table 4. Category counts for train, validation, and test data

Later, we created data loaders to iterate over the batches during both the training and validation phases. We performed a grid search to optimize hyperparameters. During our experiments, we used different learning rates, batch sizes, numbers of epochs and maximum length of the tokenized text. For the grid search, we used ARNES HPC, which helped us to speed up the process.

During fine-tuning, we used the validation dataset to evaluate the model. This helped us to determine the best model and to avoid overfitting. At the end of each different hyperparameter setting, we evaluated the model on the validation set. We measured many metrics such as accuracy, precision, recall, F1-score and more. To select the best model, we used the F1-score, which is a harmonic mean of precision and recall. The best results were achieved with the following hyperparameters: learning rate: $5e-5$, batch size: 16, number of epochs: 4 and maximum length of the tokenized text: 128. The best results are listed in the table 5.

We evaluated the performance of the fine-tuned model by testing it on the testing dataset. We separated the testing dataset from the beginning to avoid any data leakage from the training and validation datasets. We also measured many metrics, the same as in the validation dataset. We got similar results as in the validation dataset, which means that the model is not overfitting (Table 5). We think that the model is well-trained and can be used for the classification of the text.

In the end, we would like to compare our Accuracy with the accuracy of the model that always predicts the most fre-

Dataset	Accuracy	Precision	Recall	F1 Score
Validation	0.7272	0.6752	0.7272	0.6911
Testing	0.7025	0.7200	0.7025	0.6952

Table 5. Results of the fine-tuned model

quent class. The most frequent class in our dataset is "Seminar" with 333 samples or around 55% of the dataset. If we always predict the most frequent class, we would get an accuracy of 0.55. So this means that we have meaningful results.

We will also try to analyze why the model is not performing better for some messages.

- Prediction: Deliberation, True label: Seminar
  - Message: "Me too I had to reread the first couple pages several times before I understood what was going on."
  - Analysis: This misclassification likely occurred because the model mistook the reflective and discussion-oriented nature of the message.
- Prediction: Deliberation, True label: Procedure
  - Message: "Perfect. 'See' you then"
  - Analysis: The brevity and informality of the message likely confused the model. While it seems like setting up a meeting (procedure), the model interpreted it as a deliberation.
- Prediction: UX, True label: Deliberation
  - Message: "At the top of the screen are two questions, I believe that is what the 1 & 2 are on the bottom portion."
  - Analysis: This message involves navigation, which the model likely associated with user experience. Clarifying the context of instructional guidance as part of a deliberation would likely improve the model's accuracy.

Llama 2

We also implemented (source [5]) text categorization using the Llama 2 model. The first part of the implementation was similar to the previous one. We used the same dataset and split it into training and testing sets. This time, we converted the training texts into prompt texts with instructions for the model. The prompt text was as follows:

```
Categorize the following text into one of
the following categories:
1. Seminar
2. UX
3. Procedure
4. Social
5. Deliberation
6. Imaginative entry
7. Other

Input:
How do you think it should end, Emilie?

Response:
Seminar

END
```

The test data was also converted into prompt texts with instructions for the model but with empty responses. We trained the model using the training data on ARNES HPC and evaluated the model. We obtained the following results: an evaluation loss of 0.560 and a perplexity of 1.75.

That is where the problem started. We attempted to use the model to predict the categories of the test data, but while running the code on ARNES HPC, we encountered an issue. The model stopped working either after training or during evaluations. The program became stuck and did not produce any output. We attempted to run the code multiple times, but the problem persisted. Consequently, we were unable to evaluate the model on the test data.

Results comparison and human annotated data

We also self-annotated the dataset (3 different people) and selected the most frequent class as the correct class. Comparing the results of the self-annotated dataset with the original dataset, we got an accuracy of 0.686. The more important comparison is the one with the BERT model, which has an accuracy of 0.7. This means that the BERT model is better than the self-annotated dataset. There are some cases where the BERT model and the self-annotated dataset made the same incorrect predictions. For example, message "Hey guys I got through the next page" and also "Good idea." were both predicted as "Social", since they are both short messages and the first one includes greeting. However, the correct label is "Seminar". Another example is "Submitted" which was predicted as "Procedure", since it contains a task that was accomplished. However, the correct label is "Deliberation". So, because of the short messages, the model didn't have enough information to predict the correct label.

We gathered all the results of accuracy in the table 6.

Method	Accuracy
TF-IDF	0.6033
BERT	0.7025
Human annotated	0.686

Table 6. Results of the fine-tuned model

Future directions and ideas

Our results using BERT were good, but there is still some space for improvement:

- **Fixing the Llama2 model**: The Llama2 model was not working properly. By debugging and fixing these issues, we can ensure that the model operates as intended. Once fixed, we can fine-tune the model and use it for the classification task. Potentially this could lead to better results than the BERT model.
- **Using a different model**: Exploring other models for our classification task could lead to better results. By experimenting with different architectures, we can identify which model offers the best performance for our specific task.
- **Fine-tuning the BERT (or other) model with more data**: Although we used the pre-trained BERT model, there is potential to enhance its performance by fine-tuning it with additional datasets that are more closely related to our classification task. This process involves training BERT on new, relevant data to help it learn specific patterns and features pertinent to our task, thereby improving its ability to classify our data accurately.
- **Active learning**: Active learning is a powerful technique that can help improve the model performance while reducing the amount of labeled data required. Initially, the model is trained on a small subset of the data. Subsequently, it identifies and is retrained on the most challenging examples, which are the instances the model is most uncertain about. By focusing on these difficult cases, the model can learn more effectively, leading to improved overall performance.
- **Evaluation on different Datasets**: Evaluating the model on different datasets would provide better insight into its overall performance. This can show some weaknesses and areas where the model may need improvement.

These improvements offer various pathways to enhance the classification task, potentially leading to more accurate and reliable models and results.

References

[1] Man Lan, Chew Lim Tan, Jian Su, and Yue Lu. Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):721–735, 2009.

[2] Chau Minh Pham, Alexander Hoyle, Simeng Sun, and Mohit Iyyer. Topicgpt: A prompt-based topic modeling framework, 2023.

[3] Lingyao Li, Zihui Ma, Lizhou Fan, Sanggyu Lee, Huizi Yu, and Libby Hemphill. Chatgpt in education: A discourse analysis of worries and concerns on social media. *Education and Information Technologies*, Oct 2023.

[4] Shraddha Anala. Text classification using tf-idf, 2020. Accessed: 2024-05-21.

[5] Kshitiz Sahay. Fine-tuning llama 2 for news category prediction, 2023. Accessed: 2024-05-21.