**FRI Natural language processing course 2021**
**Project report**

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

# IMapBook Collaborative Discussions Classification

Luka Bračun, Klavdija Veselko, Demian Bucik

**Abstract**

In this paper we explore natural language processing approaches that aim to classify replies from discussions into predefined categories. Classes range from content discussion, greeting, logistics to feedback, response and others.

**Keywords**

natural language processing, collaborative discussion classification, online chat

*Advisors: Slavko Žitnik*

## Introduction

Natural language processing (NLP) is a field of research where artificial intelligence, computer science and linguistics meet. Text classification is one of the NLP applications. It is defined as the process of categorizing free text according to its content. In this project we will address the text classification of collaborative discussions in online chat. For testing data we will use conversations from IMapBook [1], a web-based technology that allows reading material to be intermingled with interactive discussion. IMapBook users have access to a chat and text box where they collaborate to formulate an answer to a given question. Each message in the testing data was manually annotated with some classes, based on the information in the message. The goal of our project is to build a classifier, that would annotate messages with these classes. Such classifier could then be implemented into this platform, and could help with keeping pupils focused on discussion about the book.

## Related work

What can we do with natural language processing? We can find answers on specific questions, offensive language detection, chatbots, autocorrect and autocomplete,... The goal of our project is to classify messages (short text) from chat. Short texts compared to documents have less contextual informations, meaning they are more ambiguous, which poses a great challenge for short text classification. Examples of short texts are tweets, chat messages, reviews, search queries,...

The most used vector representations of words that capture well the semantic information are Word2Vec [2] and GloVe [3]. Both models learn vector of words from their co-occurrence information (how often they appear together in large text corpora) and are pre-trained. Word2Vec is algorithm that uses neural networks to produce word embeddings and it does not have explicit global information embedded in it by default. GloVe is an unsupervised algorithm and has a global co-occurrence matrix with an estimate of the probability that a certain word will appear together with other words.

There are also contextual embeddings such as ELMo and BERT. They assign each word a representation based on its context, thereby capturing uses of words across varied contexts and encoding knowledge that transfers across languages [4].

Some research work was already performed specifically using data obtained from IMapBook systems. According to [5], of particular interest was the use of NLP techniques for automatic grading of open ended questions. Specifically, the research questions that were tackled were which NLP algorithms could provide more accurate feedback to open ended inference question while using a minimal amount of data, and how well can such techniques be used across multiple languages.

## Data

Discussion dataset consists of 3 tabs. The first one includes (crew) chat data from pupils, that were discussing answers from books. Each row contains information about the course they were attending, book id, topic (question), bookclub, pseudonym (pupil ID), message content, code preliminary, message time, yes/no if message is the answer (all no), page (shows on which page the activity was shown) and response number, which links us to the grading in third tab of the dataset. Third tab has data about grading and final answer pupils submitted. Data in second tab has the same format as the data in the first one. The only difference between these two is, that the second tab only contains discussion data, without answers and without grades.

There is a total of 712 rows in the crew data part of the dataset. We are interested in the code preliminary classes and there's 16 of them. As we can see from Figure 1 the distribution of classes is skewed, which might present additional challenges during model fitting. This can be especially problematic when the distribution in the training set differs from the distribution seen in the real world. Luckily for us, training and testing sets will be constructed from the same data and will have similar distributions.

By far the most common class is content discussion, followed by response and logistics. The bottom half of the classes are practically non-existent.

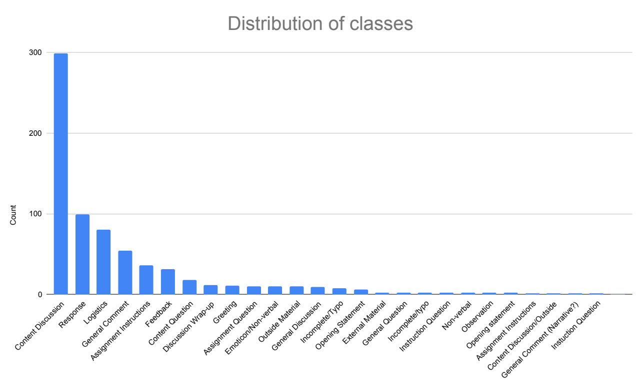


**Figure 1.** Distribution of classes in the provided dataset of collaborative discussions.

Data was collected from 6 different courses, with EDF6284 P2 having the highest number (229) of chat messages, and EDF6284 P1 with the lowest number with only 13 messages. 48 pupils participated. Average length of a message is 11 words. Figure 2 shows the number of messages each user sent in a chat, while figure 3 shows the word count for each user accross all their messages. Based on this we can conclude that most active user was edf-15 with 104 messages including 1118 words, while the least active user was dig-07 with only one message including 3 words.

Pupils were distributed in 26 groups. 10 of those groups either consisted of only one person, or only one person was active in the group. 8 groups consisted of two persons and remaining 8 groups consisted of three persons. The two persons that were most active (edf-15 and edf-16), were both in the same group. While edf-16 contributed the same amount of messages as edf-15, edf-15 contributed way more words to the conversation and consequently more words to the final response. 435 words from edf-15 messages and 250 words from edf-16 messages were included in their final response.

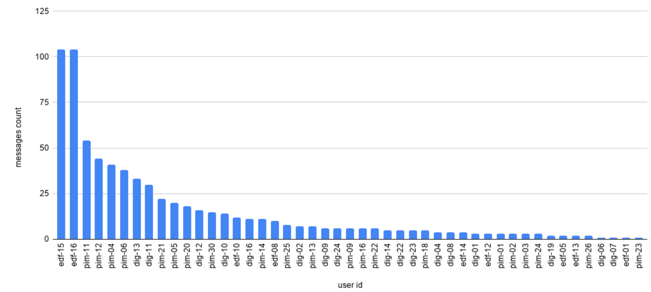


**Figure 2.** User activity: number of messages users sent in chat

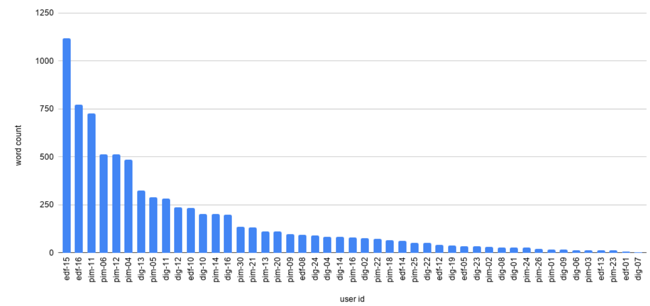


**Figure 3.** User activity: total word count across all messages per user

## Preprocessing

Since there were a few target classes with only one or two representatives, we decided to group them with the most appropriate classes. Having a class with a single representative does not make a lot of sense, since you can not evaluate the performance of the model. Considering that we use 4-fold stratified cross validation for the model evaluation, we wanted each class to have at least four representatives to simplify model training. Accounting for classes not present in the dataset would make training much more tedious without added benefits. Additionally, it seems that messages were inconsistently and somewhat carelessly annotated, and one could argue for many cases the another class would be more appropriate. Such as a message

- :)

that was tagged as a *Response* instead of an *Emoticon/Non-verbal*. Or the message

- /add

which was tagged as an *Assignment Question* instead of an *Incomplete/Typo*. Additionally, observe the following two messages.

- to mask, or not to mask. to eat, or not to eat. indoors, outdoors. out of state visitors.

- The man who moves a mountain begins by carrying away small stones.

They both fit in the context of the surrounding discussion similarly and sound alike. The first example was labeled as *General discussion*, and the second as *General comment*.

In the following paragraphs we try to justify some of our choices for grouping.

For example, the class *Observation* has only 2 representatives. We joined it with the *Feedback* class. Let us first look at a few messages from the *Feedback* class to get the general feel of how the replies sound:

- Oooo I like that idea of how he might send her to the arena and then tell her which is behind each door.
- Flipping a coin may not be the worst idea, honestly!
- That's a creative idea!

Now we can compare them with two examples of the *Observation* class:

- It feels collaborative in a way that the previous discussion was not, for me.
- I think this is turning into a message board...

We observe that examples annotated as *Observation* do sound like *Feedback*, so the joining makes sense.

Next let us take a look at a few examples of messages annotated as *Outside material*

- https://www.army.mil/article/125327/Army_designing_next_generation_protective_mask/
- Perhaps the stands they sit on can be made into mosaics like Gaudi's parks in Spain.

Comparing the with the two *External material*, we notice that one could use either of the two label for some of these replies.

- This isn't directly in the article, ...
- ...but Teresa made me also think that one way to invest in our future is to recruit, support, and mentor designers from marginalized communities with perspectives most mainstream designers don't have.

We decided to map the two examples from the *External material* to the *Outside material* class.

In total we grouped 11 classes, but most were due to typos, for example joining *Opening statement* and *Opening Statement*, or *Non-verbal* and *Emoticon/Non-verbal*, or *General Comment (Narrative?)* and *General Comment*. In the end we were left with 15 distinct classes.

### Books

We preprocessed the books that students were answering the questions about. First we removed the special characters that were not recognised by the punctuation and converted all letters to lowercase. We also lemmatized/stemmed the words with the help of NLTK and removed the stop words. Table 1 shows us top 10 words from each book. Book 1 in table presents the book with title "Design for the future when the future is bleak", Book 2 "Just have less" and Book 3 "The lady or the tiger". "wa" appears in all three books as a top 3 word. It originated obtained from words such as was and were, but stemming only kept the root.

| Book 1 | Book 2 | Book 3 |
|---|---|---|
| design | wa | wa |
| future | maier | door |
| wa | bag | king |
| world | said | one |
| one | bottega | person |
| designer | one | arena |
| city | put | would |
| different | veneta | upon |
| people | table | opened |
| like | elevator | princess |

**Table 1.** 10 most common words in each book.

Book 1 consists of a total of 2579 words, Book 2 1744 and Book 3 2532. After lemmatization and stop words removal, we calculated that Book 1 has 1085 unique words, Book 2 has 733 and Book 3 has 816 unique words.

## Methods

In this section we present a few methods which we explored in our task of classifying messages.

### TF-IDF

TF-IDF stands for *Term frequency - Inverse document frequency* and is a statistical measure that evaluates how relevant a word is for a specific messages in a collection of messages. It is frequently used for information retrieval tasks. We will use it to vectorize messages and feed them to other models, such as *Logistic regression*.

TF-IDF is calculated by multiplying two quantities

$$tf-idf(w,m,D) = tf(w,m) \cdot idf(w,D). \tag{1}$$

Where $tf(w,m)$ is the frequency of the word $w$ in the message $m$, calculated as the number of o times the word occurs in the message, divided by the number of all words in the message. And $idf(w,D)$ is an inverse document frequency of the word $w$ in the set of all messages denoted as $D$. It measures how much information the word provides. It is calculated as

$$idf(w,D) = \log \frac{|D|}{|\{m \in D : w \in m\}|}, \tag{2}$$

where $|D|$ is the number of documents or messages, and the denominator represents the number of messages where the word $w$ appears.

### Logistic regression

Since we are trying to predict multiple classes, we will actually use the multi-class version of *Logistic regression* also known as *Softmax regression*.

It is a generalized linear model that assumes a logit link function and a categorical distribution of the dependent variable. Specifically, for a single data point $(x,y)$ it assumes the distribution

$$p(y=j\,|\,X=x) = \sigma(\theta_j^\top x) \tag{3}$$

$$= \frac{e^{\theta_j^\top x}}{\sum_{i=1}^{M} e^{\theta_i^\top x}} \tag{4}$$

where $\sigma$ is the softmax function, also known as the inverse of the logit link function, and $\theta_i$ are the model parameters. We assume that $x_0$ is always equal to 1, and $M$ is the number of classes. Using the maximum likelihood estimation or maximum aposteriori estimation and assuming data points are independent and identically distributed, we arrive at the categorical cross-entropy loss function, which can be minimized by multiple optimization algorithms.

### GloVe

*GloVe* [3] stands for *global vectors for word representation* and is an approach to non-contextual word embedding that aims to capture semantic relationship between words in their embeddings. First a global co-occurrence of words is constructed, based on a large text corpus, where the element in the $i$-th row and $j$-th column is the probability $P_{ij}$ that word $j$ occurs in the context of the word $i$. The probabilities are simply estimated from the text as normalized counts of co-occurences that sum to one for each word. This probability can be written as

$$P_{ij} = \frac{X_{ij}}{X_i}, \tag{5}$$

where $X_{ij}$ is the count of co-occurence and $X_i$ is the count of word $i$.

GloVe model then tries to model the relationship in the following way

$$\log X_{ik} = w_i^\top \tilde{w}_k + b_i + \tilde{b}_k, \tag{6}$$

where $b_i$ and $\tilde{b}_k$ are biases, $w_i$ is the embedding of the word $i$, and $\tilde{w}_k$ is a separate context embedding of the word $k$.

From this goal the weighted least squares cost function is constructed and optimized. Luckily the co-occurence matrix is very sparse, which makes its construction and this optimization feasible.

### BERT

BERT (Bidirectional Encoder Representations from Transformers) is a Transformer based technique for natural language processing developed by Google. The Transformer is a model that uses mechanism of weighing the influence of different parts of the input data. There are two steps in the framework: pre-training and fine-tuning. During pre-training, the model is trained on unlabeled data over different pre-training tasks. For fine-tuning, the model is first initialized with the pre-trained parameters, and all of the parameters are fine-tuned using labeled data from the downstream tasks. Each downstream task has separate fine-tuned models, even though they are initialized with the same pre-trained parameters [6].

### Custom features

To improve performance results we also used custom features. The aim of these features was to determine what properties affect on determing certain classes. We used the following features:

- Char Count: Number of characters in the message.
- Book Similarity: Number of words in the message that appear in the book. Based on *book_id* we lemmatized the words from the specific book and compared them to the lemmatized words from the message.
- Word Count: Number of words in the message.
- Uppercase Count: Number of uppercase letters in the message.
- Question Words count: Number of appearances of ”what”, ”when”, ”who”, ”where”, ”why”, ”which” and ”how” words in the message.
- Contains Link: Checks if message contains a link.
- Contains Lmoticon: Checks if message contains an emoticon.

## Results

Results using 4-fold cross validation as are shown in Table 2. We used *negative log loss*, *f1 macro* and *accuracy* to measure each the model's performance. Metric *f1 micro* is in our case equivalent to *accuracy* so we decided to not include it. Comparing first only models that only use messages to make predictions, glove and bert performed almost equally well, but were outperformed by TF-IDF and logistic regression by a fair margin. Based on this preliminary result, we decided to only focus on improving the logistic regression based model with inclusion of custom features. We can observe that custom features substantially improved performance. Additional explanation will be provided the next section. For both bert and glove models we used pretrained word embeddings.

| model | neg_log_loss | accuracy | f1_macro |
|---|---|---|---|
| majority | -18.358 | 0.468 | 0.043 |
| glove | -8.493 | 0.631 | 0.343 |
| bert | -1.444 | 0.637 | 0.366 |
| tfidf-logreg | -1.244 | 0.666 | 0.419 |
| tfidf-cfeat-logreg | -1.083 | 0.696 | 0.464 |

**Table 2.** Classification metrics obtained with 4-fold cross validation. Majority model serves as a baseline. Glove and bert models were outperformed by logistic regression using TF-IDF features. Since logistic regression yielded the best results, we decided to upgrade it with custom features and create the model tfidf-cfeat-logreg, and that significantly improved previous results.

### Custom features

We scaled all custom feature to equal variance and used them together with TF-IDF features as an input to logistic regression. The Table 3 presents maximal logistic regression coefficient values for custom features and the corresponding classes. Maximal coefficient value tells which class each feature helps predict best. Additionally, we can see which features were most helpful in addition to TF-IDF. Results are mostly not surprising but still quite interesting. For example book similarity feature and well as message length indicate content discussion label. And since TF-IDF does not incorporate the length of the message, it makes sense that it was really useful.

However, it is fair to mention we did not eliminate correlated features before model fitting. An obvious such example would be the word and character count.

| Custom Feature | Max Coef. | Predictive Class |
|---|---|---|
| Char Count | 3.03 | Content Discussion |
| Book Similarity | 2.56 | Content Discussion |
| Word Count | 1.20 | Assignment Instructions |
| Uppercase Count | 1.12 | Greeting |
| Question Words Count | 1.10 | Content Question |
| Contains Link | 0.69 | Outside Material |
| Contains Emoticon | 0.49 | Emoticon/Non-verbal |

**Table 3.** Logistic regression maximal coefficients values for custom features that were scaled to equal variance, and the corresponding predictive classes. Custom features were combined with TF-IDF features during training.

## Discussion

In a way it is surprising that a simple logistic regression with TF-IDF features already outperformed deep models. Although that does make sense, since messages are very short, and there is not a lot of data, simple models tend to work well, and complicated models with hundreds of thousands of trainable parameters tend to quickly overfit. Although we are sure it would be possible to improve the deep models if we created our own embeddings or later on carefully fine tune the models. But since for example a single evaluation of the bert model took hours, we were not able to perform many experiments.

Using custom features we were able to improve the basic TF-IDF model. During preprocessing we made sure to separate punctuation and made it available to TF-IDF as a separate token. When constructing custom features we were looking to incorporate features that were not already included by TF-IDF, such as the message length and book similarity.

Checking all custom features coefficients and their corresponding classes, we were able to obtain some quite intuitive and interesting results that were discussed in the Results section.

In the future it would be interesting to see whether a model that is able to model feature interactions, such as gradient boosted decision trees, would be able to significantly outperform logistic regression. Additionally, it would be intriguing to know at which size of this specific dataset deep models would start outperforming traditional ML models.

## References

[1] Grandon Gill and Glenn Gordon Smith. Imapbook: Engaging young readers with games. *Journal of Information Technology Education: Discussion Cases*, 2(1), 2013.

[2] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[3] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[4] Qi Liu, Matt J Kusner, and Phil Blunsom. A survey on contextual embeddings. *arXiv preprint arXiv:2003.07278*, 2020.

[5] Glenn Gordon Smith, Robert Haworth, and Slavko Žitnik. Computer science meets education: Natural language processing for automatic grading of open-ended questions in ebooks. *Journal of Educational Computing Research*, 58(7):1227–1255, 2020.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.