IMapBook Text Classification (work in progress)

Žiga Babnik, Miha Štravs, Kristian Šenk
University of Ljubljana
Faculty of computer and information science
Večna pot 113, SI-1000 Ljubljana
zb1996@student.uni-lj.si, ms8816@student.uni-lj.si,
ks3803@student.uni-lj.si

Abstract

In this paper, we will be focusing on text classification on book discussions. We will be given many primary school students discussions and we will try to determine when the teacher should intervene (either they got the answer or they drifted too far from the subject). We will classify sentences based on book relevance, sentence type (question, answer or a statement) and based on different categories and sub-categories. For classification, we will try many different models and at the end pick the best one (or some combination of them).

1 Introduction

The task of our paper was to classify text to predict book relevance, type and category.

We were given three Slovenian stories and discussions of the stories by primary school students. Based on their discussion we would like to decide when it is time for the teacher's intervention.

The problem with interpreting the human language is that it is not a set of rules or binary data that can be fed into the system, from which we could understand the context of a conversation. With new algorithms and powerful processors, we have made a significant advancement in Natural Language Processing (NLP). NLP has been around for quite some time for languages like English, but for Slovene, it was popularized in recent years. Slovene is one of the harder languages to learn, firstly since it contains not only singular and plural word forms but also dual word forms and secondly, it contains declension which can cause many problems even for humans. One of the main challenges in the case of the given data set will be the use of slang.

In the Methods section [3] we will first look at data that we will later process and classify. In the next subsection, we will look at the ideas of our implementation [3.2]. First we will look at the preprocessing [3.2.1], than vector representation [3.2.2] and at last at classification [3.3]. For classification we will try many different classifiers. Currently we use only logistic regression and support vector classification for testing the data we preprocessed.

2 Related work

Text classification is a well researched field, the main topics discussed in papers related to text classification are word representations and machine learning models used for prediction.

The field of word representations has in recent years seen many breakthrough technologies emerge. In the paper *Efficient Estimation of Word Representations in Vector Space* (Mikolov et al., 2013) two models for word representations are presented, these models still achieve reasonable results nowadays. First the *Continuous Bag-of-Words model* (CBOW) is introduced, which uses the neighboring words as the input for prediction of the current word, after which the *Skip-gram model* (Skip-gram) is introduced, which uses the current word to predict the neighboring words. A newer approach is presented in the paper *Bag of Tricks for Efficient Text Classification* (Joulin et al., 2016a), where a model named *fasttext* is introduced. The *fasttext* model is similar to the CBOW model, while allowing for very fast training on large data sets. One of the newer approaches to construct word representations is presented in the paper *Deep contextualized word representations* (Peters et al., 2018), which introduces the *Embeddings from Language Models* (ELMo) representation. ELMo representations are contextual, deep and character based, allowing the representation to take into account syntax, semantics and how the uses of the words vary across linguistic contexts.

Related work is presented much more thoroughly in survey paper *Text Classification Algorithms: A Survey* (Kowsari et al., 2019), where various word representations and machine learning models, such as the ones presented in the previous paragraph, are used and compared using many evaluation methods.

3 Methods

3.1 Data

We were given data collected from an online discussion forum, on which primary school students were able to discuss and comment on three different books, that they previously read. Each discussion entry contains various annotations about the date and place of the entry, user who submitted the entry and actual label that represent the classes for the given text classification.

3.1.1 Basic data statistics

For the fist classification task, where the aim was to find entries relevant to the book discussion two labels were given, the values of which are shown in table 2, while the distribution is shown in table 1.

Label	Count
No	2155
Yes	1384

Table 1: Book relevance value distribution

Book relevance	
Label	Meaning
YES	Value is relevant to the book.
NO	Value is not relevant to the book.

Table 2: Book relevance explanation

For the second classification task, where the aim was to predict the type of entry, we were given three labels. The explanation of each label is shown in table 4, while the distribution is shown in table 3.

Label	Count
Statement	1710
Answer	1155
Question	672

Table 3: Entry type tag distribution

For the final classification task, where the aim was to predict the category of the entry, five labels were given, where each label was further divided into sub-labels. The explanation of the label and sub-label is shown in table 5, while the distribution is shown in table 6.

Entry type	
Label	Meaning
QUESTION	Entry is a question.
ANSWER	Entry is answer to any question.
STATEMENT	Entry is sentence that is not question or answer.

Table 4: Entry type explanation

3.1.2 Class dependence

Since the data includes three classes, we are interested if they are independent. If it turns out there exists some dependence we could use a prediction of one class to improve the prediction of a different dependent class. Since *Book Relevance* and *Type* are quite simple in regards to number of tags, we would like to see if they could assist in predicting the *Category* class.

Firstly we present the prior distribution of the *Category* class in figure 1. We see that the distribution is unbalanced, with the *CO* and *DA* label being most frequently represented, while some other labels have close to zero appearances. From this we could construct a classifier that only guesses the top four or five labels and achieve fairly good results.

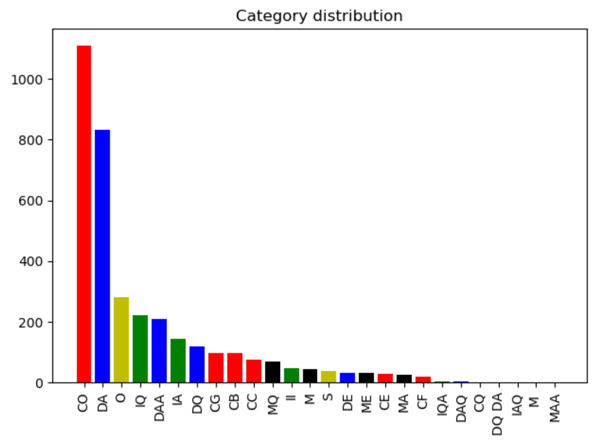

Figure 1: Descending order of distribution of labels for the *Category* class, where each *Category Broad* label is assigned a unique color

We observed how the *Category* distribution changed when we first split the data into all com-

Category		
Label	Sub-tag	Meaning
CHATTING	CG	Greeting
	CB	Chatting about books
	CE	Encouraging others to join the chat.
	CF	Chatting about how they feel.
	CO	Chatting about other thing.
	CC	Being mean
SWITCHING	S	Talking about where in the system they currently are.
DISCUSSION	DQ	Discussion question.
	DE	Posing a question that directly encourages further discussion.
	DA	Answering the discussion question directly.
	DAA	Answering a question or commenting on the answer of someone else.
MODERATING	ME	Encouraging the students.
	MQ	Asking questions relevant to the discussion question.
	MA	answering the discussion question directly
	DAA	Answering the students questions
IDENTITY	IQ	Identity question
	IA	Identity answer
	IQA	Combination of previous tags
OTHER	O	anything that does not make any sense (typos ...).

Table 5: Category label and sub-label explanation

binations of *Tag* and *Book Relevance* labels. Figure 2 shows the results. While quite a few com-

Label	Count
Chatting	1430
Discussion	1197
Identity	416
Other	282
Moderating	177
Switching	38

Table 6: Category tag distribution

binations of labels prove to be useless since they are underrepresented, some provide useful information for classifying the *Category* class. Among the most informative are *NO+S* meaning the message is not relevant to the book and is a statement, the messages of this kind are most likely to represent the *CO* category label, *Yes+A* meaning the message is relevant to the book and represents an answer, the messages of this kind are most likely to represent the *DA* category label. Both of these results make sense when talking about the meaning of the *Category* class labels, they most likely represent.

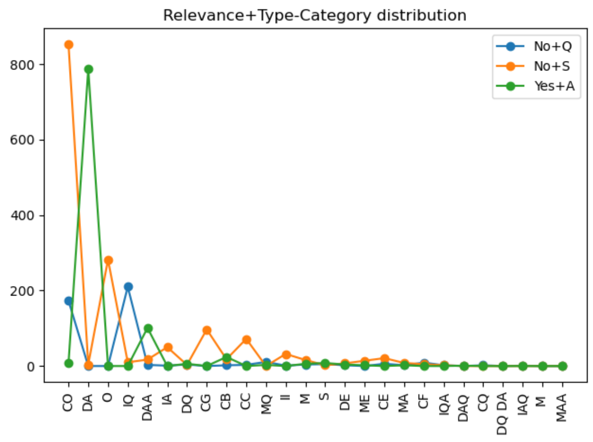

Figure 2: Descending order of distribution of labels of the *Category* class, where the data was first split by all *Book Relevance* and *Type* class label combinations after which the splits containing less than 10% of all data were removed

From these results it is clear that the classes *Book Relevance* and *Type* contain information about the *Category* class.

3.1.3 Category class sequence dependence

The data is presented as a sequence of messages each with different class labels. As such we are interested if there are any class label sequence patterns which could assist us in classification. First we divided the data by *Book Clubs* and *Topics* and than sorted all comments by their time stamps. From the sorted messages we were able to extract pairs of labels, where the first label in the pair is followed by the second label. By constructing a Markov chain where all labels represent nodes and all pairs represent links between these nodes, we can visualize the sequence dependencies. Furthermore we can count the number of all pair occurrences and encode them as link weights, normalizing them allows us to represent the probability of the second label following the first label. For the final step we remove all links that occur less than 1% of times compared to all links in the Markov chain, and all isolated nodes, this way the final network will be more clear and present only the most important information.

The figure 3 shows the Markov chain of sequences of the *Category* class where we take into account only one previous label. From the results, we can determine that most labels are followed by that same label, while the probabilities of switching between tags is somewhat lower for most labels. To take into account more of information we construct a network where we take into account two and three previous labels.

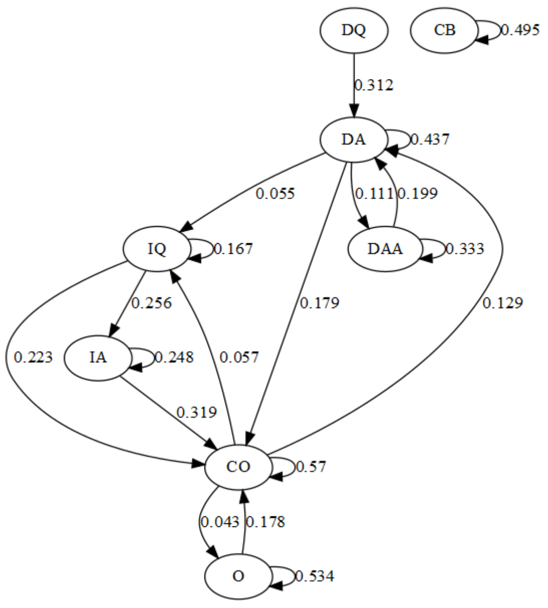

Figure 3: Markov chain of the *Category* class sequences, where we take into account only one previous label, with normalized weights and with links that occur less than 1% of the times removed

Figures 4 and 5 show Markov chains constructed when we take into account two and three previous labels respectively. To generate the network when taking into account three previous labels we lowered the minimal weight for which the link is kept from 1% to 0.5% of all sequences. We see that when we already observe two or three of the same labels the probability of observing the same label again is quite high.

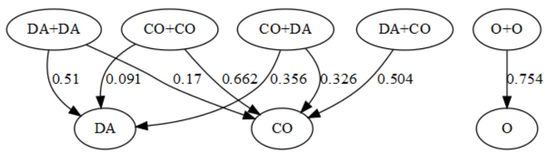

Figure 4: Markov chain of the *Category* class sequences, where we take into account two previous label, with normalized weights and with links that occur less than 1% of the times removed

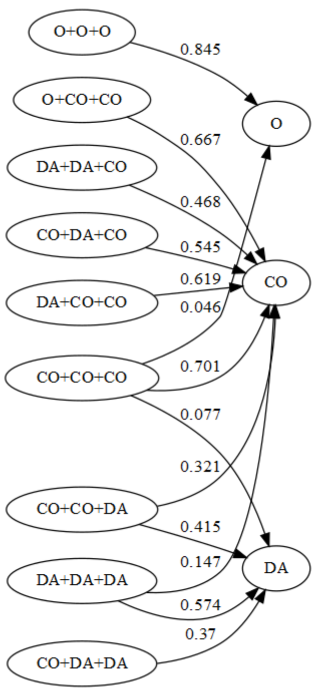

Figure 5: Markov chain of the *Category* class sequences, where we take into account three previous label, with normalized weights and with links that occur less than 0.5% of the times removed

3.2 Current work

We introduce the work done so far.

3.2.1 Data preprocessing

We first applied basic preprocessing steps on the given data. Using tokenization we extracted individual words from sentences, added tags and multiplied words that define interrogative sentences such as *who* and *why*. Both steps were performed using the reldi-tokeniser and reldi-tagger described in the paper *Multilingual Text Annotation of Slovenian, Croatian and Serbian with WebLicht* (Ljubešić et al.). Since we are dealing with texts containing informal language we didn't remove any stop words.

Tokens were then transformed into vectors using bag of word approach or with word embeddings. Tags were filtered so that only selected few remained. They were then vectorized using bag of word approach and appended to the message vector. Topic was also appended using bag of word approach. Lastly for category classification we used book relevance which was predicted before category prediction and added to the vector in a form of number -1 or 1.

3.2.2 Vector representation

As mentioned in *Bag of Tricks for Efficient Text Classification* (Joulin et al., 2016b) the use of word vector embeddings can boost the performance on problems with small amounts of data. We represented each sentence by summation of individual word vectors of a sentence, which were represented in the form of bag-of-words (BOW), word2vec embeddings and Elmo embeddings.

We used contextual embeddings proposed in the paper *High-Quality ELMo Embeddings for Seven Less-Resourced Languages* (Ulčar and Robnik-Šikonja, 2019), since context holds valuable information for our classification tasks. For comparison we also used fast text embeddings trained on wikipedia pages found on their webpage. Since we are dealing with informal text, mistakes might be a common occurrence, which is why We used the pymagnitude library from *Magnitude: A fast, efficient universal vector embedding utility package* (Patel et al., 2018) for querying the embeddings, since it will make a similar vector even in the case of typos. It also includes lazy loading which is more RAM friendly.

3.3 Classification

After preprocessing we can start with classification. We input given vectors from preprocessing into one of the classifiers. Classification can be done using multiple algorithms and different classifiers and we will try a few of them and see which one (or some combination of them) gives us the best results.

4 Future work

1. We used normal fasttext embeddings and Elmo embeddings which are which are word based embeddings. We might get some boost by using Bert embeddings which are sentence based.

2. As mentioned in subsection 3.1.3 there may be some information hidden in the sequence of messages. Therefore our model might improve if we add category and broad category of previous three messages.

3. Since we might lose some information by simply adding word vectors to represent a sentence, we might consider a different approach, where we concatenate all word vectors in the sentence.

4. Data analysis uncovered a few interesting statistics, which are useful for classification. We might further investigate the data to uncover any new possible statistics that may improve classification.

5. Currently we use a single model to classify the Category class. It might prove useful to first train a classifier that distinguishes between CategoryBroad and than further train separate classifiers for each label of CategoryBroad.

5 Results

For categories and broad categories we have chosen two benchmarks. F1 micro which shows us which embeddings gave as an overall better precision and F1 macro which favors ones with better representation of smaller classes.

5.1 Logistic regression

We can see in table 7 that BOW had best overall results while fasttext were beter for fitting cases from less represented classes. Elmo embeddings took the middle ground.

5.2 Support vector classification

In table 8 we can see similar but worse results when comparing to logistic regression. Fasttext wiki vectors did achieve overall best micro F1 compared to logistic regression.

Word vector model	micro	macro
	Category	
Bag of words	0.647	0.346
Fast text wiki	0.567	0.38
High-Quality Elmo	0.619	0.362
	Broad Category	
Bag of words	0.74	0.61
Fast text wiki	0.7	0.6
High-Quality Elmo	0.722	0.627

Table 7: F1 scores for book category classification with logistic regression

Word vector model	macro	micro
	Category	
Bag of words	0.618	0.301
Fast text wiki	0.566	0.396
High-Quality Elmo	0.556	0.36
	Broad Category	
Bag of words	0.71	0.53
Fast text wiki	0.662	0.56
High-Quality Elmo	0.651	0.558

Table 8: F1 scores for book category classification with support vector classification

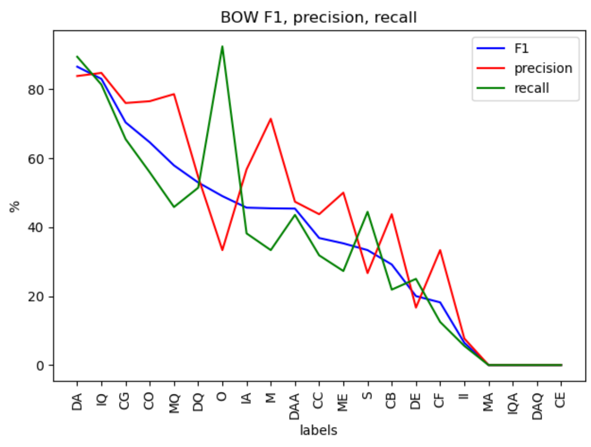

Figure 6:

6 Discussion

Discuss the results and provide possible further research questions.

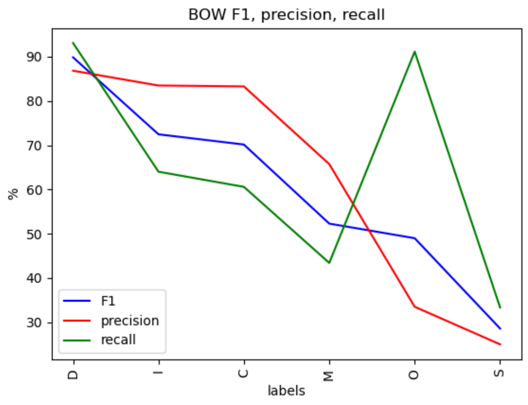

Figure 7:

References

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016a. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016b. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

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

Nikola Ljubešić, Tomaž Erjavec, Darja Fišer, Erhard Hinrichs, Marie Hinrichs, Cyprian Laskowski, Filip Petkovski, and Wei Qui. Multilingual text annotation of slovenian, croatian and serbian with weblicht.

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

Ajay Patel, Alexander Sands, Chris Callison-Burch, and Marianna Apidianaki. 2018. Magnitude: A fast, efficient universal vector embedding utility package. *arXiv preprint arXiv:1810.11190*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Matej Ulčar and Marko Robnik-Šikonja. 2019. High quality elmo embeddings for seven less-resourced languages. *arXiv preprint arXiv:1911.10049*.