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 from the most basic ones as Naive Bayes classifiers [3.3.1] to more advanced ones as deep learning. [3.3.4].

2 Related work

Present related work.

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 tags that represent the classes for the given text classification.

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

Tag	Count
Yes	1384
No	2155

Table 1: Book relevance value distribution

Book relevance	
Tag	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 tags. The explanation of each tag is shown in table 4, while the distribution is shown in table 3.

Tag	Count
Question	672
Answer	1155
Statement	1710

Table 3: Entry type tag distribution

Entry type	
Tag	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

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

3.2 Idea of implementation

We propose the idea of our implementation.

3.2.1 Data preprocessing

We will first apply basic preprocessing steps on the given data. Using tokenization we will extract individual words from sentences, each token will then be lemmatized. Both steps will be performed using 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 will be removing only a small portion of possible stop words. The information if the text directly refers to the main question data will also be added. This is important as it shows the flow of the conversation and can be then also used to check if the post is referring to the forum question.

3.2.2 Vector representation

As mentioned in *Bag of Tricks for Efficient Text Classification* (Joulin et al., 2016) the use of word vector embeddings can boost the performance on problems with small amount of data. Therefore we will represent each sentence in the form of a vector using word vector embeddings. We will use

Category		
Tag	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 tag and sub-tag explanation

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

Table 6: Category tag distribution

contextual embeddings since context holds valuable information for our classification tasks. We will most likely be using embeddings proposed in the paper *High-Quality ELMo Embeddings for Seven Less-Resourced Languages* (Ulčar and Robnik-Šikonja, 2019). For comparison we will also use 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 will be using 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 with a small typo. 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.

3.3.1 Naive Bayes classifier - Work in progress

One of the basic ones is naive Bayes classifier.

Naive Bayes classifier is based on this equations, where c represents class and d represents document.

$$p\left(c|d\right) = \frac{p\left(d|c\right)p\left(c\right)}{p\left(d\right)}$$

The great thing about naive Bayes is that you can see what the classifier is doing in the background and you can tweak the parameters to better suits your needs. But as the name implies, this is a naive method, because it makes a very strong assumption, that two features are independent from each other, which in reality doesn't hold on.

3.3.2 SVM - support vector machine - Work in progress

One option is using support vector machine which is based on representing each word in a multidimensional space. SVM constructs a hyperplane or set of hyperplanes in a multi-dimensional space, which can than be used for classification. One of the main problems of using SVM is that it works like a black box, so tweaking parameters is almost impossible.

3.3.3 KNN - k-nearest neighbors - Work in progress

An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer).

3.3.4 Deep learning

In the recent advancement in Machine Learning, Deep Learning with the help of Neural Networks has become popular option for text classification.

4 Results

Present results produced by implementation.

5 Discussion

Discuss the results and provide possible further research questions.

References

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

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.

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*.

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