Text classifications using IMapBook dataset

Domen Kos, Timotej Kovač
University of Ljubljana
Faculty of computer and information science
Večna pot 113, SI-1000 Ljubljana
dk6314@student.uni-lj.si, tk3713@student.uni-lj.si

Abstract

In this paper we discuss various approaches when trying to classify messages from an IMapBook dataset. We use some of the traditional nature language feature extraction methods which are then used to train some of the basic machine learning models like Naive Bayes, Logistic Regression and Support Vector Machine. We design and test this features by following standard machine learning principles. Trained models are later used to determine the time when a teacher should intervene in the conversation that is taking place if this would drift off topic.

1 Introduction

For the course assignment at *Natural language processing* class we decided to tackle with text classification problem on IMapBook dataset. The dataset contained short discussions between primary school students that were chatting about different book topics. Each record was annotated with 16 attributes. Original messages were posted in Slovene language, but they were also translated into English. We also had the information about topic they were discussing, if their message was an answer to some previous asked question and if their discussion was relevant to the topic, since there were no constraints so they could wrote anything they wanted. If the discussion was moving away from the proposed topic, the teacher should intervene and guide it back by asking some questions relevant to the book. Dataset contained approximately 3500 messages about 3 different short stories. Our goal was to develop the models which could detect the topic of the current debate so the teacher could intervene. The idea was to develop different models and combine their outputs. Our first idea was to analyze separate messages and define their relevance to the topic. We would also have to define type of message which could either be an answer or a question along with the message category. By combining results of separate messages we could determine when the conversation started to move away from the topic.

2 Related work

Text classification is a popular topic of natural language processing (NLP) field, thus there are many researchers working on it. In this section we present most relevant work and techniques they used.

Most of the research work and the state-of-the-art results were achieved using English language, but some of the techniques and approaches can be adapted to Slovene language. The authors of paper (Fernández Anta et al., 2013) had similar problem, where they analysed Spanish tweets which are also relative short texts. They discuss different approaches for preprocessing data to extract the most relevant features which are later used for classification. Few standard approaches are discussed like how to define a basic term which are used by classification algorithms. Such as unigrams (1-grams), bi-grams (2-grams), tri-grams (3-grams), n-grams. They found out that having n larger than 3 does not improve results. They also tried combining different types of n-grams (like uni-grams and bi-grams) to achieve better results. That also means that attribute list was larger so they removed some entries by setting a threshold value. With threshold they removed n-grams that did not appear frequently enough or they appeared to many times, since they were considered as noise. They also discuss how important are stemming and lemmatization comparing English and Spanish language which is also interesting for Slovene language which is also morphological richer then English.

In second paper (Lai et al., 2015) authors presented the recurrent convolutional neural network for text classification. The network was able to capture contextual information of the sentence and extract features and learn some word representation. As described the disadvantage of the recurrent network is that although the context of the word is captured, the model is biased where later words are more dominant then earlier. To tackle this problem they applied additional convolutional and pooling layers. They learned a word representation and used it for some text classification.

Another similar approach is used in third paper (Johnson and Zhang, 2014) where the authors used convolutional neural networks for text categorization where the word order was also taken into account. The input to the network is not standard bag-of-word representation but they present their own word representations which are some higher dimensional vectors where 2D convolution is applied. For baseline model they used a support vector machine (SVM) classifier with bag-of-word representation and showed that their approach gave them lower error rate than standard models.

There are also many already pre-trained word representations which can be used, also for Slovenian language. The word embeddings induced from a large collection of Slovene texts composed of existing corpora of Slovene were prepared and published on CLARIN (Ljubešić and Erjavec, 2018). This could also be useful with our task since the embeddings were learned on bigger corpuses then are available to us.

3 Initial ideas

To determine if the teacher must intervene we need to answer the following questions:

- are the messages book relevant,
- what type is the message,
- in what category does it belong.

Based on this information we could then determine if the conversation is in need of an intervention or not.

Because there are three seperate requirements our first idea was to come up with three seperate classifiers. We would start with standard text classification procedures like tokenizing, stemming, removal of stop words and then represented words as vectors in order to use them in our machine learning algorithms. After that we would probably have to use some kind of machine-learning approach. Recurring convolutional neural networks or SVMs could be used to make use of the sequential information of words as well.

3.1 Book relevance

Here the answer we are trying to answer is whether the message is related to a story or not. From the data itself came to some conclusions:

- Category of the message is a good indication whether the message is book relevant. So if the message is classified as having a category discussion it is a good chance that the message is book relevant. So the result of the message category classifier could be used here to determine if the book is relevant.

- Conversations have some retention. If the conversation starts leaning towards a discussion of a book most messages will be about the book, and if the conversation starts to move towards some other category most of the messages will follow. So here the sequence and previous states could be deemed important.

A good idea would be to include the original texts from the three books that our gathered messages are referring to. Another possible approach to classify book relevance would be to use the result of the message category (see section Message category) in order to get better results.

3.2 Type of the message

Here we try to determine the type of the message. It can be a statement, a question or an answer. Here too we drew some conclusions from the data available:

- Answers tend to follow questions. So message order is important.

- Answers are mostly regarded as book relevant and statements are not.

Detecting questions can be done by adding a feature that checks if the sentence contains a question mark or some sort of question word. To determine answers from statements we will most likely have to take into account the order of sentences as well.

3.3 Message category

Each message can be one of the following:

- **chatting**, which doesn't fall into any of the bellow categories;
- **switching**, which mostly consists of asking for help;
- **discussion**, which consists of some particular keywords ('lahko', 'bi) and descriptions of activities, objects;
- **moderating**, where the teacher leads the conversation and which we could identify by maybe checking for grammatically correct sentences;
- **identity**, where we could check the appearance of question marks or question words or
- **other**, where there is mostly gibberish.

Some manual features could be added which would be especially efficient for determining the categories **other** and **identity**. Very long words or those that contain a single repeating character can quickly be classified as "other" as well as those that only contain emojis or other special characters. Detecting question marks, question words and possibly personal names can largely contribute to classifying message as being of type identity.

4 Methods

In this section we present the pre-processing steps we performed and all the experiments we performed.

4.1 Text pre-processing

We followed some standard text pre-processing steps. First we tokenized each chat. After that we used lemmatizer to get the basic forms of each token. Both of the steps were first performed using the standard NLP tools which are available mostly for English language. Since Slovene language is a bit different the tokenization and lemmatization were not always correct. For that reason we used a tool that was designed and trained for Slovenian language (Ljubešić, 2020). Using it we obtained better token representation of our texts which were later also lemmatized using the same tool. Lemmatized tokens were then used to build different vector representations (e.g. Tf-idf) which were used to train our models. We also found a stop word dictionary for Slovene language but when removing the stop words the performance in all models dropped. We assume that the reason is that most of the texts are very short and thus after removing the stop words we end up with even smaller set of words which can not be used to successfully train the classifier.

4.2 Additional features

With the help of FeatureUnion class in the sklearn library we added our own feature extractor that was combined with a TfidfVectorizer. We first added all of the relevant stories to the training set, from which we have removed all of the special characters which greatly improved our results. After that we checked the messages and their appropriate tags and came to some conclusions and predictions that we predicted to be of some value. We then checked how these features improved the classification results and retained only those that provided good results. All of the additional features can be seen in Table 1.

classification	**custom feature**
book relevance	length of the word
	longest repeatance of a chararacter
type	contains question words or a question mark
category	length of the word
	longest repeatance of a chararacter
	contains discussion words
	contains identity words

Table 1: Manually extracted features

4.2.1 Book relevance

For book relevance we checked for some characteristics that mainly separated chat messages from book relevant ones. We added three features which were observed to be important when determining book relevance.

When children talked about a book they changed their style of writing so that certain words started to appear and they avoided writing gibberish.

The first idea was to check the length of the longest word. As Slovene does not have really long words we determined that the length of the word that was over 12 characters in length should be noted.

Next was number of repentance of a character. Again in Slovene this doesn't appear often but it does in the dataset when the messages are either gibberish or the conversation has moved to a more relaxed level and with that generally isn't book relevant.

The last feature was to check for the presence of the word 'lahko' which generally referred to book relevant messages. Because of the nature of the questions presented to children most of their sentences included the word 'lahko'. This however should not be included if the style of the questions presented to the children would change. So this last feature is only useful if the questions retain the same structure.

4.2.2 Type

To check the type of the question we only added one feature that seemed to give good results and that was to include whether any question marks or question words appears in the sentence. To determine answers from statements we would have to take into account the order of messages and couldn't find any features that could differentiate between the two.

4.2.3 Broad category

Here we included the first two features already mentioned in subsection 'Book Relevance'. Like before the longest repentance and length of the word seemed to provide good indications that the message is either about chatting or other.

Another feature was that for identity. Here the keywords 'kdo', 'jaz', and 'ime' seemed to mainly appear in. This can however be useful without overfitting to training data as these words are often used in identification scenarios. Checking for personal names and persons nicknames could also provide an improvement when trying to identify this category.

The last feature that we added was to check for the presence of words 'lahko' and 'bi', which often indicated that the message was of the type discussion. Again as already mentioned in subsection 'Book Relevance' this feature should also be taken with a grain of salt as it does help with identification of discussion related messages but only because the questions posed to the children were formatted to provide answers or messages with these words.

5 Results

5.1 Models

Initial goal was to detect when the teacher needs to intervene into the discussion. To achieve it we defined several classification models. First set of models was developed to classify each chat into one of the two classes. Either the chat is somehow relevant to the book or not. The other set of models was more complicated. They were classifying the chats into 6 groups. Each group represents a simple description of what the text is talking about. All the categories are described in the separate document.

The models we used were *Naive Bayes*, *Logistic Regression*, *Support Vector Machine (SVM)*. Each model was trained on 2653 random examples and tested on the rest 1062. The distribution of relevant and non-relevant text is plotted on Figure 1 for training set (left) and testing set (right). We evaluated them by calculating the accuracy, recall, precision and F1 score. The majority classification accuracy of predicting the relevance into the non-relevant class was 0.619.

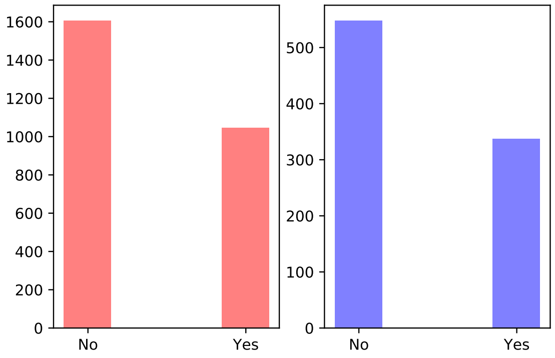


Figure 1: Distribution of relevance on train and test set

All the results from the models we tried are shown in Figure 2.

5.1.1 Naive Bayes

The simplest model that we tried was *Naive Bayes* which assumes that words in the sentence are independent.

Although the model is naive, the results we got are not that bad. The AUC of the model for predicting the relevance of the text is 91%.

	Accuracy	Precision	Recall	F1
NB	0.81	0.80	0.66	0.73
	0.50	0.70	0.50	0.56
	0.46	0.64	0.46	0.49
LR	0.85	0.84	0.75	0.80
	0.73	0.73	0.73	0.72
	0.76	0.77	0.77	0.76
SVM	0.86	0.84	0.79	**0.81**
	0.74	0.75	0.74	**0.73**
	0.78	0.78	0.78	**0.78**

Table 2: Evaluation of models

5.1.2 Logistic Regression

The performance is similar to *Naive Bayes*. But we go the AUC of 92% which is the highest of all baseline models.

5.1.3 Support Vector Machine

Overall SVM performed the best so we will use it in further experiments. We also plotted the ROC curve for predicting the relevance of the text which is shown on Figure 2.

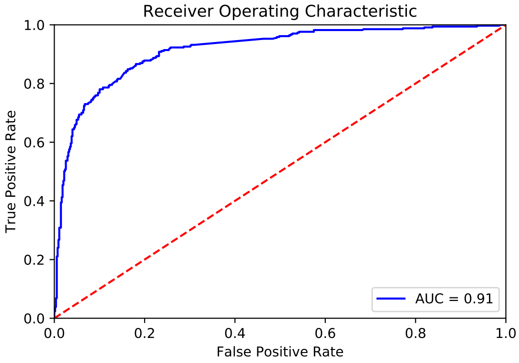
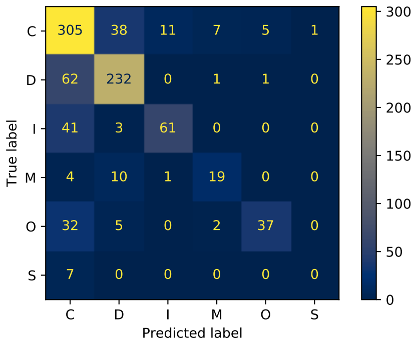


Figure 2: ROC curve of SVM performance

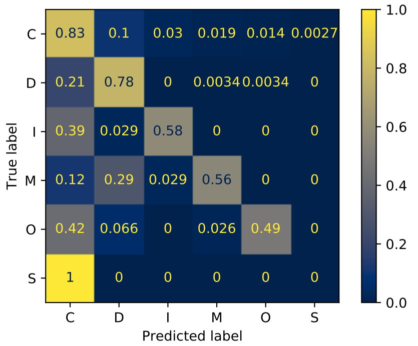
To demonstrate the performance of the SVM model when classifying the *Category* we plotted two confusion matrices where can we can observe the distribution of the testing set into the different categories and how our SVM model predicted them which is shown on Figure 3. On Figure 4 we plotted the same normalized distribution.

From the Figure 4 we can observe that most of the test examples are from category *C (Chatting)* and we are able to correctly classify 83% of them. We can also observe that 10% of the examples are wrongly classified as category *D (Discussion)*. The examples from second category *D* are correctly classified in 78% of cases. In 21% of cases the model wrongly classify it as category *C*. category *I (Identity)* is correctly classified in 58% of the cases. Most of the wrong classified examples are again putted into the category *C*. For the category *M (Moderating)* we can correctly predict 56% of the true examples. Almost 30% are wrongly classified into the category *D* and another 12% into category *C*. In the category *O (Other)* we correctly classify only 49% of the examples. Most of them (42%) are wrongly classified into the category *C* . In the case of category *S (Switching)* we wrongly classify 100% of the examples into the category *C*. From the Figure 3 we can observe that in this case there were only 7 testing examples.



True label \ Predicted label	C	D	I	M	O	S
C	305	38	11	7	5	1
D	62	232	0	1	1	0
I	41	3	61	0	0	0
M	4	10	1	19	0	0
O	32	5	0	2	37	0
S	7	0	0	0	0	0

Figure 3: Confusion matrix of classifying the Category



True label \ Predicted label	C	D	I	M	O	S
C	0.83	0.1	0.03	0.019	0.014	0.0027
D	0.21	0.78	0	0.0034	0.0034	0
I	0.39	0.029	0.58	0	0	0
M	0.12	0.29	0.029	0.56	0	0
O	0.42	0.066	0	0.026	0.49	0
S	1	0	0	0	0	0

Figure 4: Normalized confusion matrix

We can assume that our model correctly predicts most of the examples from the category *C*. We are also quite sure that the examples from the category *D* are correctly classified (with probability of 0.78). For other categories the probabilities are lower (around 50%) except for the *Switching* category which where our model fails completely. We can also observe that most of the wrong predicted labels are classified into the category *C*.

5.2 Custom features impact

Some of the custom features for classifying messages into a particular group worked better than others. In table 3 the improvements for our classification models can be seen with the included custom features.

classification	model	improvement in F1
book relevance	NB	+1,3 %
	LR	-0,2%
	SVM	+0,1%
type	NB	-0,2%
	LR	+0,3%
	SVM	+0,2%
category	NB	-0,5%
	LR	+1,2%
	SVM	+1%

Table 3: Performance improvement in F1 score with used custom features.

As it can be seen we achieved the best results when using the custom features when trying to classify message based on the broad category. There our score was improved by a little more than 1% for the LR and SVM models. This was not expected as well defined features for the type classification problem were expected to provide the best results. There an improvement was bellow 0.5% and after further investigation we identified that the problem wasn't so much in wrongly classifying questions as it was in wrongly classifying answers and statements.

Using custom features for classifying book relevance did show an improvement in NB F1 score by 1.3%. When tackling with this problem we saw a much greater improvement in our F1 score when we included the provided stories into the training data as defined in the subsection "Additional features", as we did with our manual features. Even though the words 'lahko', 'bi', 'da' often appeared in messages that were labeled as book relevant they were most probably already detected as of great value to determine the type by our models and so didn't provide much better results.

5.3 Detection of conversation drift

To detect when the teacher needs to intervene we trained two models. They were trained using the lemmatized texts as before. But in this case we took sequential texts to train and test our models. For training we took first 70% of lemmatized texts and calculate their Tf-idf vectors using our custom features. From the experiments above we decided to train SVM model for classification of relevance of the text and Linear Regression (LR) for classification of category. Both of the models were trained and evaluated using the 5-fold cross validation. The accuracy and F1 score of SVM model were 0.83 and 0.82 respectively. For the LR model we obtained 0.65 accuracy and 0.67 F1 score.

The last 30% of conversations we used to detect the drift. The idea was to take batches of sequential messages and for each we predicted the relevance label. We counted the number of relevance chats in a batch using the following method. If the label was positive ('Yes') we counted it as relevant. If the label was negative ('No') we also predicted the category. We defined some soft categories which also count as relevant. If the category was any of those we classified chat as relevant otherwise as non-relevant. From the frequency of relevant/non-relevant chats we classified each batch. The drift was detected if there were non relevant discussion in few consecutive batches.

To determine the number of sequential messages and the number of consecutive batches we manually extracted some features from the training set. First we calculated how many sequential messages are non relevant before the teacher had to intervene into the conversation. We averaged the results over the whole training set and got the average number of 4.8. This mean that when there was no relevant discussion between the participants the teacher on average intervene after 5 non relevant messages. We were also interested how often are relevant messages and how many non relevant chats are between two relevant messages. The number we got was 2.5 but since we are not interested about the consecutive relevant messages we corrected this number. We defined that if we observe more then 3 sequential relevant messages we conclude that this is a relevant conversation and thus there are no non-relevant messages in between. The corrected average we got was 4.0. Meaning that during two relevant messages (which are not marked as relevant conversation) there are on average 4 non relevant. Since we are also only 78% sure that when we predict the category *D* it is a correct label we increased both of these numbers for 50%. We defined the size of the batch as the average messages before the teacher intervene which is increased by 50% and thus we get 6. The number of consecutive batches is defined from the corrected average of non relevant chats between relevant ones. By also increasing it for 50% we get 7 consecutive batches. Since 30% of messages from the category *M* is also wrongly classified as category *D* we define two soft categories *M* and *D* which are both seen as relevant.

We notify the teacher that the conversation moved away from the topic if there is no single relevant message in the 7 consecutive batches of 6 sequential messages.

6 Discussion

We achieved quite good results using the traditional natural language processing approaches. We mainly focused on implementing and refining these and have gotten satisfying results. Our best results were with the use of the Support Vector Machine classifier equipped with custom features gave us F1 scores of 0.81, 0.73 and 0.78 for the classification task of determining book relevance, category and type of message respectively. Our implementation also gave us good results when tackling the main problem of detecting the conversation drift. We trained two models using our manually extracted features. Here we had to determine some additional parameters. By manually inspecting the dataset and calculating some specific properties we were able to to correctly determine the sizes of batches and the number of messages in the batch. We found out that the results very much depend on how good we are familiar with the data we use and how many distinguish features we are able to extract. There may be better approaches but even with this one we were able to quite successfully and reliably detect when the conversation started to move away from the original discussion so we could inform the teacher about it.

We also started experimenting with more advanced models where features are learned and we do not need to manually extract them. Since we were limited by time and hardware the results were not comparable to the ones we got by manually extracting features.

7 Conclusion

We were successfully in implementing various classifiers for determining book relevance, category and type of messages and by combining them we build a good detector for conversation drift. Although we could have gotten better results using more advanced methods and models we rather chose to focus our efforts on improving the results using more traditional machine learning models and by manually extracting different features depending on the problem we were solving. Future directions are to train some deep neural models. Manual feature extraction takes long time and is not as abstract as it could be. These manual features contributed to improving the category classification but didn't perform as well for the type or book relevance classification. For the following more work would have to be done to come up with better features. We also need to be very familiar with our data and thus we design features that are consistent with our dataset which is another bias to our model. With deep learning approaches we could build a network that would be able to learn this feature on it own and they would be less biased than current ones.

References

Antonio Fernández Anta, Philippe Morere, Luis Chiroque, and A. Santos. 2013. Sentiment analysis and topic detection of spanish tweets: A comparative study of nlp techniques. *Procesamiento de Lenguaje Natural*, 50:45–52.

Rie Johnson and Tong Zhang. 2014. Effective use of word order for text categorization with convolutional neural networks. *CoRR*, abs/1412.1058.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

Nikola Ljubešić. 2020. The CLASSLA-StanfordNLP model for lemmatisation of standard slovenian 1.1. Slovenian language resource repository CLARIN.SI.

Nikola Ljubešić and Tomaž Erjavec. 2018. Word embeddings CLARIN.SI-embed.sl 1.0. Slovenian language resource repository CLARIN.SI.