A Novel Automatic Features Extraction Approach for Text Corpora

Ettore Randazzo
University Of Illinois At Chicago
Department of Computer Science
eranda5@uic.edu

Benedetto Vitale
University Of Illinois At Chicago
Department of Computer Science
bvital3@uic.edu

Abstract

In this paper we address the task of automatic supervised feature extraction, with the aim of minimizing the number of features while maximizing their relevance. Other methods generally produce a huge number of features, sometimes most of them not so relevant w.r.t. the task in question, also disregarding the information brought by the document labels. We present a novel supervised feature extraction algorithm that extracts features of various kinds, usable either with single and multiple speakers documents and multiple labels classification. We perform various tests with different corpora to show the robustness and scalability of the features extracted. We also perform comparisons with a state of the art basic feature extraction method, showing similar results with an order of magnitude less features extracted. Finally, we discuss eventual future works to improve the quality and variety of the features extracted and to automatically fine-tune the algorithm for various different tasks.

1 Introduction

In fields like data mining, machine learning and natural language processing, features are a quite crucial aspect. Anybody working in any of these fields knows that features are fundamental in order to guide a classifier toward the right prediction, given a specific task. In particular, engineering specific features for a certain dataset requires a considerable effort. However, there are also cases in which adopting a reasonable number of standard features (that could be generated through an algorithmic procedure) is going to provide more than reasonable results, or in a more general case, would be an important baseline onto which add other, more complex and domain specific features. The field of Natural Language Processing(NLP) could be seen as one of those, for example for some specific tasks such as sentiment analysis and supervised topic modeling. For these tasks there are indeed many automatic features extraction algorithms providing good results, even if they usually produce a huge amount of features, which could complicate in a considerable way the computational part and prohibit the insertion of other features of different types. What we propose is an innovative features extraction algorithm with the aim of producing general features to maximize the results and at the same time minimize the number of produced features, such to ease the computational burden of having too many features as well as being able to add several more features without having problems caused by the curse of dimensionality. In this paper we show how our algorithm can be applied to different corpora, which, in our case, are a movie review one and a dialogue corpus; for the dialogue corpus in particular, we will use our algorithm both for classifying the performance and the topic. More details on the corpora will be provided in Section 3.
In addition, in order to properly show that the features generated are effective we used our features with a set of classifier in related prediction tasks, reporting their performances. We also performed a parameter analysis in order to understand how robust is our model with slight changes on the input parameters.

2 Related Work

Feature extraction is a vast area of interest and different topics regard this argument. We can reasonably subdivide its domain in two parts: finding novel features from text, and selecting relevant features out of a big number of them, possibly by creating new ones which relate to the pre-

vious ones. The latter has been studied with different methods, (Wang et al., 2003) is an example of that, which compares different famous algorithms such as LDA, PCA and SVM in order to understand when it is better to use one over another.
Extracting interesting features directly from the text is generally trickier and here as well various methods have been tried. (Broda et al., 2013) and (Bird, Steven, 2006) have presented tools that among the other things accomplish this task. However, both of them use mostly unsupervised learning algorithms which inevitably make the number of features extracted to be very high. Very famous and successful feature extraction schemes which don't consider labels are TF-IDF and Bag of Words, however, the enormous quantity of features that they extract forces the user to reduce its dimensionality in various ways, such as using one of the methods showed in (Wang et al., 2003) and removing stop words, or wisely selecting a smaller number of most frequent words. Even after that, the number of remaining features is usually still very high. (Maas et al., 2011) analyzes the results of such schemes while also introducing a new semi-supervised feature extraction method which however performs similarly to Bag of Words alone. However, we are interested in completely supervised feature extraction methods that could exploit the information given by the labels to significantly reduce the number of features extracted while maintaining their relevance for the classification task. We are not aware of any work doing that, so we decided to implement our own algorithm, inspired by the previous unsupervised and semi-supervised work.

3 Corpora

We used two different dataset for the chosen tasks:

3.1 Transcripts Dataset

The first is the 'transcripts dataset', gently offered by (NLP@UIC,), which consists in 35 dialogues related to sessions involving a tutor and a student learning basic computer science concepts for the first time.
Since we used the dataset for two different tasks, we performed two different preprocessing procedures, which are the following:

3.1.1 Dialogues Preprocessing

For the task of predicting in a binary ('POSITIVE' or 'NEGATIVE') way the performance of students, we first determined the relative label for each session by computing the average score of the student for each session. The reason why we computed the average is that there are sessions (where one session corresponds to one file) in which more than one topic is touched and the student has a score for each topic treated during one session. We believe that one session has to be considered altogheter and not split it, because the amount of time that a student had to stay focused may very well influence the performance of the learning. Thus, for estimating the final performance of a single session, after having computed the average score over all the topics touched, we checked if it was greater than a certain threshold to set the final label, which is 'POSITIVE' if greater or equal than the threshold, 'NEGATIVE' otherwise. The threshold was computed as the median of all the scores in the various sessions.
Then, given the huge amount of numbers and keywords referring to specific notes in the test of the exercises considered during the various sessions (e.g. node 'xxx' or '#', etc.), we preprocessed the dialogues by replacing all these keywords with the token 'UNK', while the numbers with the token 'NUM'.
In addition, since we are interested only in the kind of speakers (STUDENT and TEACHER in this case), for every dialogue we kept just the sentences relative to a speaker, deleting the lines marking start and end of the dialogue, as well as the sentences relative to the description of events occurring in the meanwhile. For the sentences kept we also marked the beginning of each sentence according to if it is relative to a student (STU as first word) or to a tutor (TEA as first word). Once cleaned the corpus, we then performed POS tagging with the Python library 'nltk(natural language toolkit)' and in the end separated the dialogues in positive and negative documents according to the previous labeling operation, obtaining the final dataset used to perform the experiments.

3.1.2 Topic Extraction

The second task we performed is extracting features for supervised topic classification of students sentences. In order to do this, given that we have three topics in the dialogues ('tree', 'stack', 'list'), we extracted all the sentences of students referring to a specific topic having more than three words(to ignore sentences relative to pausing or confirmation as 'uh uh' and similar) and in which there was

no explicit occurrence of the keywords (also plural forms) relative to each topic. Once extracted the sentences in three separate files, these were the relative amount of sentences left for topic:

Tree	Stack	List
812	234	434

We then stratified them by taking the first 230 for each topic for our experiments.

3.2 Movie Reviews Dataset

The second is the one used in (Maas et al., 2011), which is a dataset for binary sentiment classification containing substantially 25,000 highly polar movie reviews for training, and 25,000 for testing. We decided to use this to further test our features on the binary classification task. To do so the data in the reviews were properly preprocessed by replacing all the numbers in the reviews with the token 'NUM' and by removing the 'html' tags by means of regular expressions. Then we performed POS tagging with the same library of the first dataset. Next, we left the train and test division untouched for this dataset during our experiments.

4 Methods

Our feature extraction model is subdivided in two parts: relevant feature extraction and feature computation.

4.1 Relevant Feature Extraction

First, we feed our feature extraction algorithm with the labeled train datasets. Because of the fact that we want to extract a small number of relevant features, we need to specify different parameters in order to give thresholds of relevance.
We implemented different versions of this feature extraction algorithm, one for each type of dataset, each of them having something slightly different from the others.

4.1.1 Movie reviews classification

For the movie reviews dataset, we only have 2 classes, where in each document there is only one speaker. In this case, our algorithm will try to extract features regarding the count of general things like number of words per document, number of sentences per document, number of words per sentence. A feature will be considered relevant if the ratio of its value between positive and negative documents is higher than a user specified threshold.
Also, the algorithm will try to extract n-gram counts of words and, if specified by the user, also of POS tags. The algorithm will try to find relevant features up to a user specified number of grams, i.e if the user specifies up to 3-grams, it will look at unigrams, bigrams and trigrams. As for the general features, a feature will be considered relevant if the ratio of its value between positive and negative documents is higher than a user specified threshold, which is generally different from the threshold of the general features. Also, in this case, we want the overall count of that specific n-gram we are considering to be at least a user specific minimum count, which generally can vary for different n-grams. We do this because we are trying to extract a small number of relevant features, thus we don't want to extract features of words that appear just once or twice. This choice deserves some considerations: it is undeniable that very rare words are most likely to be very important features for classification, because they usually appear in only one of the two (or multiple as well) labeled types of documents. However this has a drawback: *they are very rare!* This implies that in order to use them, we need a huge number of rare words, so that at least one of them actually appears in the document we are looking at. We give the user the possibility to not select those very rare features, which implies losing accuracy for the gain of having several less features. Also, we stuck to these considerations throughout this paper, so we always set the minimum counts much bigger than 1, to effectively extract a small number of relevant features, by losing accuracy in the process.

4.1.2 Transcript dialogue quality classification

For the dialogue quality classification, even if we still have only two classes, we have different speakers, the teacher and the student. In this case, our algorithm (which has been generalized for any number of different speakers) looks at all the features checked by the movie reviews feature extraction algorithm, but it looks at them multiple times: at first it disregards the speakers and does exactly as done before, then it only considers the sentences of a specific speaker and tries to extract those features there as well, for every speaker. How the features are considered relevant is done exactly as

before.
Finally, the algorithm also tries to extract cross-speaker features, that is the ratio of words, words per sentence and sentences between two different speakers. If these ratios are significantly different from positive and negative documents, then we add these new features.

4.1.3 Transcript topic classification

The topic classification is very similar to the movie reviews classification. The only difference is that this time we have 3 different classes. So we generalized our feature extraction algorithm for the movie reviews to take into account any number of different classes. To do so, we decided that a feature would have been relevant in an "ANOVA" style: if the count of this feature for a class is significantly different from at least another class, then it is relevant.

4.2 Feature computation

The relevant feature extraction algorithm outputs a list of features to be computed. Then, for every dataset we have, that is train and test of every class, we run our feature computation algorithm, that, given the list of features to compute and a dataset, outputs a list of values for each document, representing the computed values of each feature. What we usually do after is write those values in a .csv file.

5 Results and Discussion

We ran different classifiers for every classification task:

5.1 Transcript dialogue quality classification

For the dialogue classification, having only 15 positive and 15 negative documents, we were able to do leave-one-out cross-validation. We tried it with many classifiers, as the training of such a small dataset was extremely fast. Table 1 shows the results.

Being it a stratified dataset, the majority class baseline of accuracy is 50%. This implies that logistic regression and naive bayes both with and without the POS features performed very poorly. The best result was achieved by SVM with polynomial kernel. Also, here POS tags appear to improve its results.

Method	Dialogue Classification w/ POS[1]	Dialogue Classification w/o POS[2]
Logistic Regression	0.5	0.5
K-Nearest Neighbors	0.6	0.56
SVM Linear Kernel	0.63	0.66
SVM Polynomial Kernel	0.66	0.7
Naive Bayes	0.53	0.56

Table 1: *Accuracy results for various classifiers w.r.t. the task of predicting performance of sessions by performing a leave-one-out cross-validation*

These results show that even with a very small dataset, our feature extraction algorithm is able to extract useful features that improve the classification. In order to consider even a different baseline, we ran the algorithm with min frequency equal to 1.1 and n-gram count 1 with the best performing classifier without using the POS features, which is SVM with a polynomial kernel, obtaining as result an accuracy of 0.8, slightly better than our best result, which is 0.7, but with 6000 features, against the 1300 used in our default case.

5.2 Transcript topic classification

For this dataset composed of 230 sentences per class, we decided to do a 10-fold stratified cross validation. The accuracy results are shown in table 2.

Here, the majority class baseline is 33%, being it stratified. As we can see, SVM performs worse than logistic regression, and POS features appear to improve its results. However, even though the best result is significantly better than the baseline, it is not outstanding. We believe it is that way because of the difficult task we aimed at. Nevertheless, it succeeds at showing that these features are relevant for improving the classification. In a similar fashion to the previous experiment, we ran the algorithm with min frequency equal to 1.1 and

[1] results are statistically different according to the ANOVA test, with a p-value of 0.0003

[2] results are statistically different according to the ANOVA test, with a p-value of 0.0004

[3] results are statistically different according to the ANOVA test, with a p-value of $1.46e^{-319}$

[4] results are all statistically different according to the ANOVA test, with a p-value of 0

Method	Topic Classification w/ POS[3]	Topic Classification w/o POS[4]
Logistic Regression	0.5625	0.519
K-Nearest Neighbors	0.425	0.4188
SVM Linear Kernel	0.478	0.504
SVM Polynomial Kernel	0.337	0.334
Naive Bayes	0.514	0.491

Table 2: *Accuracy results for various classifiers w.r.t. the task of predicting topic for sentences of students by performing 10-fold cross-validation*

n-gram count with the best performing classifier without using the POS features, which is Logistic Regression this time, obtaining as result an accuracy of 0.0.584, slightly better than our best result, which is 0.5625, but with 530 features, against the 150 used in our default case.

5.3 Movie reviews classification

Being it a very big dataset, we weren't able to do many tests, also, we did the traditional train and test subdivision, without performing any cross-validation. Results are shown on table 3.
These results are significantly better than the baseline of 50%. Also, it appears that POS features worsen the results, even if by a slight amount, so we decided to perform a t-test on the SVM prediction with and without POS to understand if they were significantly different. The results we obtained are statistically different (t-statistic=2.3905, p-value=0.01683). One reason for such a worsening might be because they are less relevant than the ones regarding words and the predictors behave worse with more features. We should perform more tests to validate this assumption.
This dataset has been vastly used among various researchers, so as it can be seen in (Maas et al., 2011), we have several results to compare ours with. A reasonable baseline is usually using the Bag of Words features, which in this case accomplishes an accuracy of 0.88. Their result is significantly outperforming ours, but that was expected, as it uses an enormous amount of features, by selecting the 5000 most frequent words appearing in the documents, compared to our model which

Method	Sentiment Classification w/ POS	Sentiment Classification w/o POS
SVM Linear Kernel	0.822	0.825
Naive Bayes	0.755	0.771

Table 3: *Accuracy results for various classifiers w.r.t. the task of predicting sentiment of movie reviews by using the provided test and train data*

only uses 435 relevant extracted features. However, our goal is to extract a very small number of features, still obtaining good results, and it is successful at it, noticing that by using only one tenth of the number of the Bag of Words features it performs only around 6% worse. Also, it is safe to assume that by testing our algorithm with various configurations of parameters we can increase even more its accuracy, while still keeping low the number of features.

5.4 Parameters Analysis

Having so many parameters to set, the quality of our feature extraction might vary a lot depending on how we actually caliber them. Thus, we believed that an analysis of the performance of the models with varying parameters could have been very interesting, in order to find some good guidelines, and to test the overall robustness of the algorithm.
So, we ran the algorithm with values for the minimum frequency ratio of n-grams in the interval [1.1,2.0], with an increment of 0.1 at every run. To perform the tests, we used the classifier which achieved the best results in the previous tables for some classification tasks we performed. We did it both for the performance classification and the topic classification tasks; we would have liked to perform it as well for the movie dataset, but the dimension of that corpus made it a really computationally hard task for our computers. The results for the performance classification task are shown in Figure 1 and Figure 2, where we used SVM with polynomial kernel as classifier.

As we can see, the best results using POS is obtained for values in the range [1.4,1.6], while without POS with 1.1 (the best results are reported in our tables). It appears that having POS features improves the robustness of the algorithm, which can be seen by the big range of values where the highest performance is obtained. Without POS,

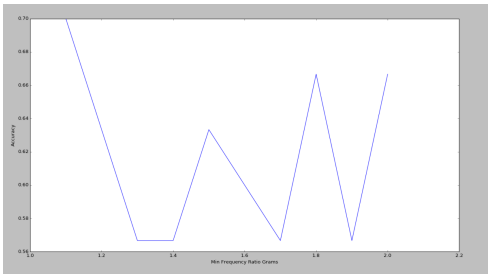

Figure 1: *Plot varying the minimum frequency for ngrams w/o POS tagging for the dialogue performance classification task*

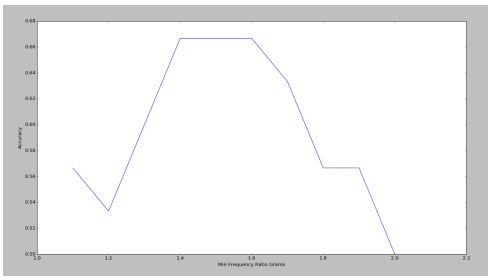

Figure 2: *Plot varying the minimum frequency for ngrams w/ POS tagging for the dialogue performance classification task*

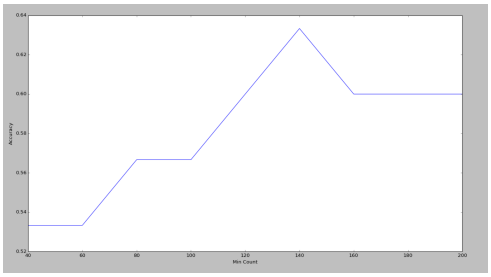

Figure 3: *Plot varying the minimum count for ngrams w/o POS tagging for the dialogue performance classification task*

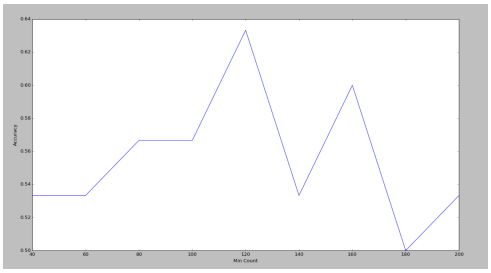

Figure 4: *Plot varying the minimum count for ngrams w/ POS tagging for the dialogue performance classification task*

the algorithm looks like more susceptible to noise, and the fact that it performs better with a lower frequency threshold might imply the need of having more features to have stable results.
Once obtained these result, we performed another test w.r.t. the minimum counts this time. We kept fixed the minimum frequency for n-grams at the best value of the previous plots and made the minimum count vary in the interval [40,200], with an increment of 20 at every run. We did it either with and without POS features. The results of this new test are shown in Figure 3 and Figure 4, where we used again SVM with polynomial kernel as classifier.

As the plots show, the best results using POS is obtained with a minimum count of 120, while without POS with 140. We can notice that in both cases we start from a low min count and increase with a monotonically increasing curve of accuracy until we find the global optimum. After that optimum it decreases in both cases. This was expected, because by increasing the min count, at a certain point we are inevitably going to cut very relevant features. Also, very rare words (of very little counts) are never considered, because we start our analysis at a minimum count of 40. If we were to consider them, the graph would most likely have 2 local optima, one for the exact number needed to consider all the rare words, and the one that we found here. It appears that setting the right min count is very important.
We also performed the minimum frequency n-gram test in the exact same way for the dialogue performance classification task also for the sentences topic classification task. The results are shown in Figure 5 and Figure 6, where the classifier used is Logistic Regression.

We obtained the best result for the POS tagging case with a minimum frequency for n-grams of 2.0, while for the case without POS with 1.7. It appears on both cases that the results improve by selecting only the most relevant features. Also, the accuracy might still improve past 2.0. This is in contrast with the previous analysis, suggesting that for more than binary classifications, we prefer having less and more accurate features with respect to a bigger, less accurate number.

6 Conclusions

Automatic effective feature extraction methods are difficult to achieve, we showed that by using the knowledge of having a labeled dataset we can automatically extract several relevant features that don't need to be filtered with some dimensionality reduction methods and improve the classifica-

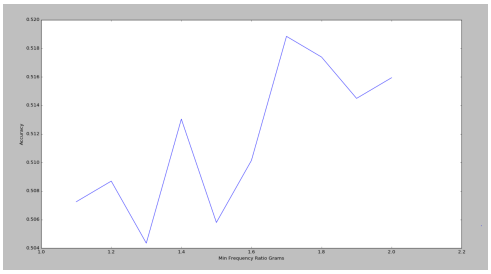

Figure 5: *Plot varying the minimum frequency for ngrams w/o POS tagging for the sentences topic classification task*

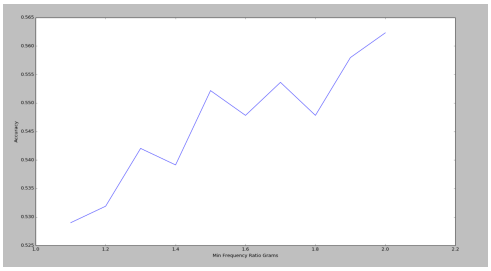

Figure 6: *Plot varying the minimum frequency for ngrams w/ POS tagging for the sentences topic classification task*

tion accuracy of any task we tried. We showed a parameter analysis to understand how crucial it is to fine tune those parameters, trying to justify with some reasoning its results. We showed comparisons between our results with Bag of Words, showing that, even if they are slightly outperformed by the latter, they require an order of magnitude less number of features.

However, fine-tuning several parameters in order to their best combination for the given task is very time consuming. Future work might want to try to automatically find the best parameters for a given dataset, or modify our algorithm in order to manually decide the exact number of features to be extracted, sorting them by relevance, thus requiring a much smaller number of parameters. Also, other types of features might be added to improve the variety of the input, and other thresholds, maybe information gain, could be considered in order to classify a feature as relevant or not.

7 Appendix

Ettore Randazzo created the Relevant Feature Extraction and Feature Computation algorithms, for all the different versions. He also performed the movie reviews dataset tests.

Benedetto Vitale performed all the preprocessing operations necessary with both datasets used. He also performed all the tests for the dialogue performance classification task and for the sentences topic classification as well.

References

Maas, Andrew L. and Daly, Raymond E. and Pham, Peter T. and Huang, Dan and Ng, Andrew Y. and Potts, Christopher, *Learning Word Vectors for Sentiment Analysis*, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, June, 2011.

Bird, Steven, *NLTK: the natural language toolkit*, Proceedings of the COLING/ACL on Interactive presentation sessions, Association for Computational Linguistics, 2006.

Wang, Xuechuan and Paliwal, Kuldip K, *Feature extraction and dimensionality reduction algorithms and their applications in vowel recognition*, Pattern recognition, Elsevier, 2003.

Broda, Bartosz and Kedzia, Pawel and Marcinczuk, Michal and Radziszewski, Adam and Ramocki, Radoslaw and Wardynski, Adam *Fextor: A feature extraction framework for natural language processing: A case study in word sense disambiguation, relation recognition and anaphora resolution*, Computational Linguistics, Springer, 2013.

Natural Language Processing Laboratory, University Of Illinois At Chicago.