CMPE492 FINAL REPORT

A SENTIMENT ANALYSIS SYSTEM

Mustafa Haluk AYDIN

Introduction

As a senior project, I chose to develop system makes sentiment analysis for a given document. According to Wikipedia[1], sentiment analysis(opinion mining) refers to use of natural language processing, text analyzing, computational linguistics and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Recently, increasing amount of research has been conducted on this topic. Since huge amount of information is available in on-line documents, obtaining some valuable information from these such as sentiments or opinions people express about a specific object is a task that attracts attention of computer scientists. Finding opinion sources, extracting related sentences, reading and summarizing them and organizing them into usable form is an extremely challenging task for a human. So, an automated system that does all is needed, namely sentiment analysis system. There are many classification types for a sentiment analysis system such as feature-based, or sentence-level sentiment classification. In my project, the system employs document-level sentiment classification methods.

Motivation

Opinions in natural language are mostly expressed in very complex manner. This makes very difficult for computer to completely understand a document written in a language. I can claim that there are many methods, algorithms, techniques waiting to be found in natural language processing because of this complexity. The motivation behind this project is to get better accuracy rate for document-level classification by applying an older algorithms modified with new features or a new algorithm designed by me. Since most existing document-level classification techniques are based on supervised machine learning, I also have employed one of them in my project. After all, this study will serve the purpose of using limited resources efficiently in this global world.

State of the Art

There are many researches that focused on sentiment analysis. I have read articles of some of these researches. In the article[2] “Thumbs up? Sentiment Classification using Machine Learning Techniques”, they also made document-level sentiment analysis and classified a document(a review) as positive or negative. For the classification, some machine learning techniques have been used, namely SVM, Naive Bayes and Maximum Entropy Classifiers. They claimed that sentiment classification is harder than topic classification, since sentiment can be expressed in a subtle manner and requires more understanding than traditional topic-based classification. For evaluating machine learning techniques, they chose to work on datasets of movie reviews which are claimed to be convenient for this task. They used an IMDb archive of movie reviews with rates which are needed for supervised machine learning methods. With different features they got results as below which express that SVM works well in this domain with 80% accuracy.

	Features	# of features	frequency or presence?	NB	ME	SVM
(1)	unigrams	16165	freq.	**78.7**	N/A	72.8
(2)	unigrams	”	pres.	81.0	80.4	**82.9**
(3)	unigrams+bigrams	32330	pres.	80.6	80.8	**82.7**
(4)	bigrams	16165	pres.	77.3	**77.4**	77.1
(5)	unigrams+POS	16695	pres.	81.5	80.4	**81.9**
(6)	adjectives	2633	pres.	77.0	**77.7**	75.1
(7)	top 2633 unigrams	2633	pres.	80.3	81.0	**81.4**
(8)	unigrams+position	22430	pres.	81.0	80.1	**81.6**

In another article[3] “Sentiment analysis using support vector machines with diverse information sources” from Mullen and Collier, the researchers introduced an approach classifying texts as positive or negative using SVMs. The approach included some diverse sources of related information. First of these sources is Turney value that is the average of all word-specific values which refer to measure of the positive or negative sentiment expressed by that word or phrase in the text. A word-specific value can be computed by subtracting PMI distance between the phrase with word ‘poor’ from the one

with word ‘excellent’ where PMI distance is calculated with some probabilities estimated by querying a search engine.

The other source is Osgood value which consists of three values, namely potency(strong or weak), activity(active or passive), and evaluative(good or bad). These values are calculated by minimal path length(MPL) in WordNet between the word(e.g. delicious) and adjectives for the value(for evaluative, good and bad). For example; potency value of delicious is MPL(delicious,good) minus MPL(delicious,bad). They used these sources with SVM to get better results. I think they reached high accuracy rate with Hybrid SVMs which combine different SVMs with different information sources.

Model	**3 folds**	**10 folds**
Pang et al. 2002	82.9%	NA
Turney Values only	68.4%	68.3%
Osgood only	56.2%	56.4%
Turney Values and Osgood	69.0%	68.7%
Unigrams	82.8%	83.5%
Unigrams and Osgood	82.8%	83.5%
Unigrams and Turney	83.2%	85.1%
Unigrams, Turney, Osgood	82.8%	85.1%
Lemmas	84.1%	85.7%
Lemmas and Osgood	83.1 %	84.7%
Lemmas and Turney	84.2%	84.9%
Lemmas, Turney, Osgood	83.8%	84.5%
Hybrid SVM (Turney and Lemmas)	84.4%	**86.0%**
Hybrid SVM (Turney/Osgood and Lemmas)	**84.6%**	**86.0%**

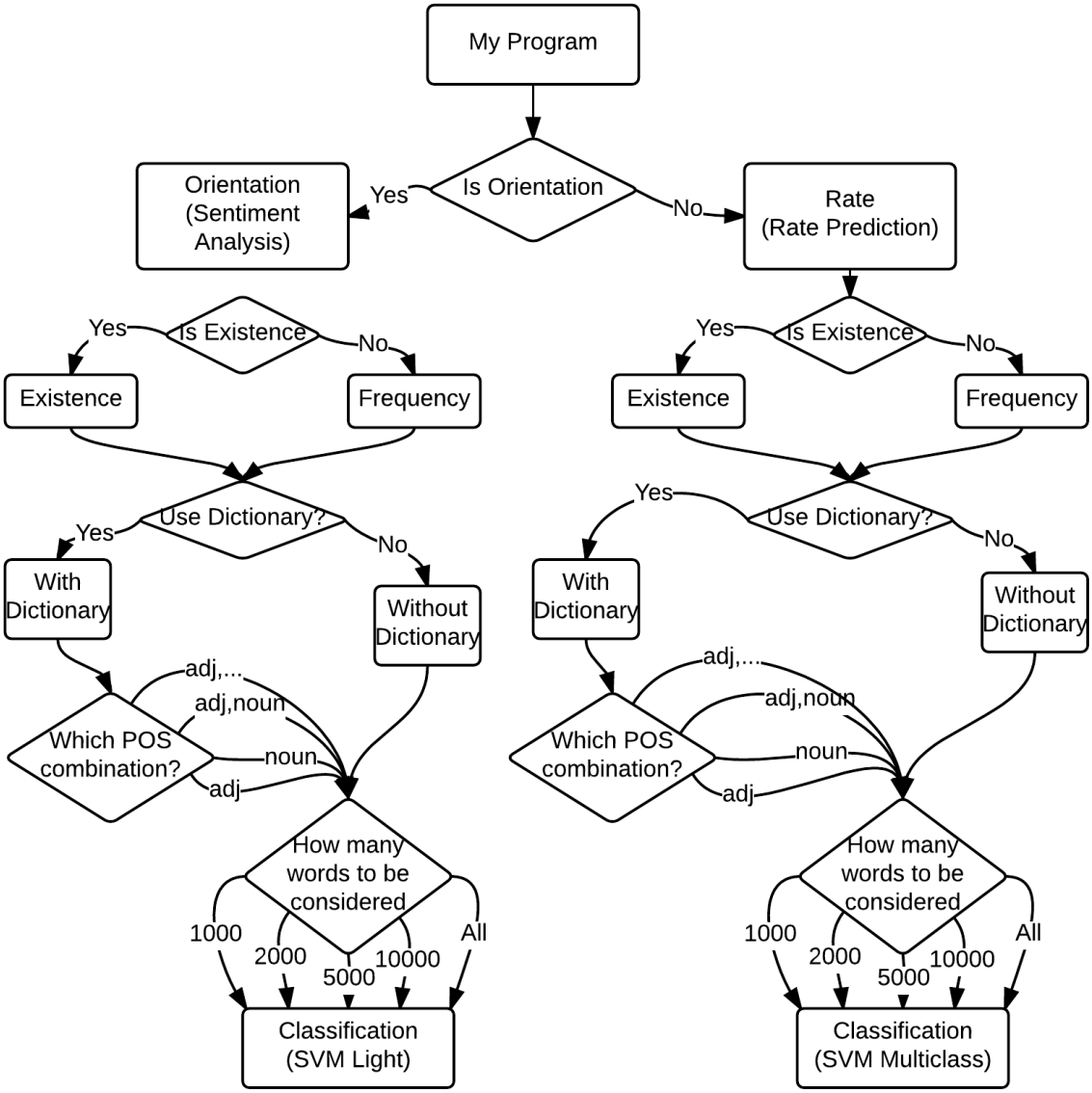
Methods and Results

In my project, I have used a supervised machine learning method, SVM. It is the best choice for sentiment classification according to some researches I examined in detail. On the other hand, I decided input type for the SVM to get highest accuracy rate. An input can include features below[8]:

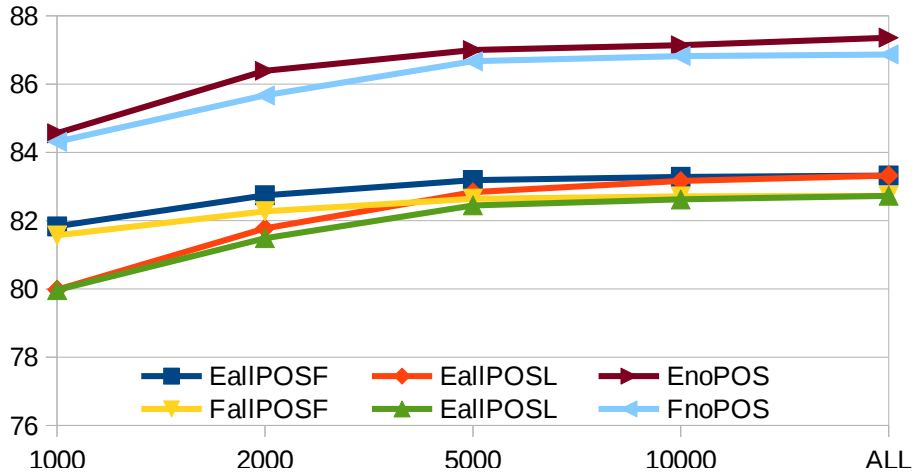
- terms and their frequencies: All words in the document, and how many times they occur in the document. This is the basic feature for any document-level sentiment classification.
- part of speech(POS): Classifying word as noun, adjective, verb, adverb, pronoun, preposition, conjunction, etc… POS can be called as lexical category or lexical class. When we do not consider POS attribute of a word, understanding sentiment in the document will be more difficult task.

I have a movie dataset from IMDb[4] which contains 25000 reviews for each positive and negative orientation. It also includes vocabulary and corresponding weights of words in

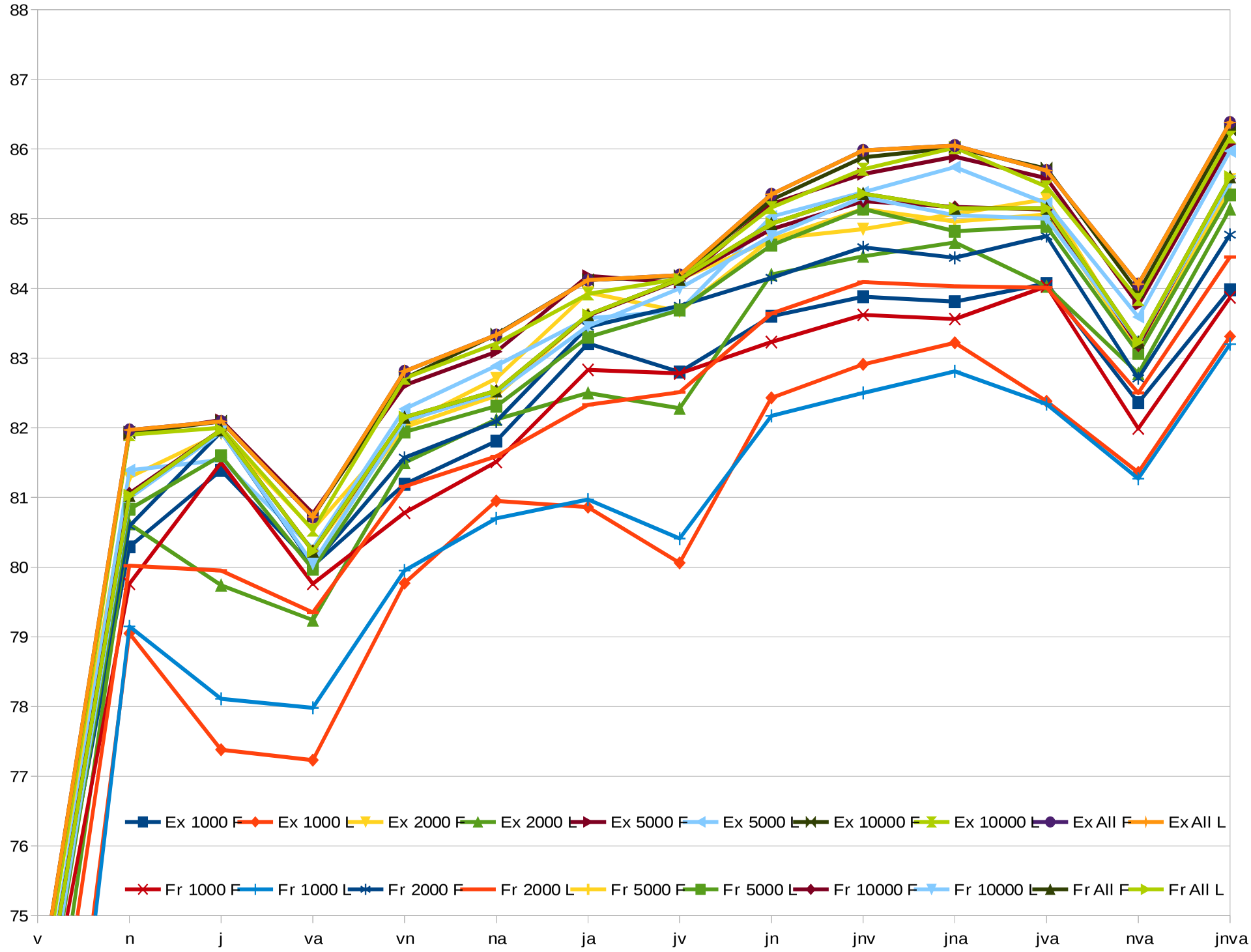
the list. Words of the vocabulary is in order according to number of uses of that word. For example, first word in the file is the most used one in the dataset. I also have a dataset, named Moby[5], which contains almost 240.000 words with their POS. I used it for POS tagging words in IMDb dataset. By using these datasets, I made a sentiment analysis which is explained in detail. I also tried to develop a rate prediction algorithm which guesses a rate from given review. The IMDb dataset also includes rate of each review. So I can use SVM for this purpose. SVM^{light}[6] is used for sentiment analysis. SVM^{multiclass}[7] is used for rate prediction. It can do multi-class classification where a rate varies between 1 and 10. The following figure demonstrates how my application work.



In my project, I have implemented both sentiment analysis and rate prediction in one program. If the program regards orientation of reviews, it makes sentiment analysis. If it regards rate of reviews, it makes rate prediction. On the other hand, the program considers existence or frequency of the words in reviews. Existence of a word means whether the word exists in the review or not. The program has dictionary option which enables the program to use POS of words. A POS can be adjective or noun or verb or adverb . Conjunction, preposition, etc … are not considered. And any combination of these POSes can be input for my program. For example, let say adjective and noun are chosen as a combination. Words that are neither adjective nor noun will be removed from the dataset. At the end, as a last option, the question of how many word should be taking into consideration is asked. The answer which is one of most important(or frequent) 1000, 2000, 5000, 10000, all words is the last input of the program. I also have a different scenario that the number restriction is applied first and the POS combination is considered later which means if I choose 1000 words and noun as a POS combination, the program will classify according to noun words in first 1000 words. I named this scenario as LastPOS and it is not shown in the figure above. Some charts for sentiment analysis(orientation) have shown below. The following figure shows the importance of number of words considered for classification. Increase in the number causes an increase in accuracy. (E | F)(allPOS | noPOS)(F | L) refers to Existence or Frequency, then average of all combination of POSes or no POS regarded, then FirstPOS or LastPOS.

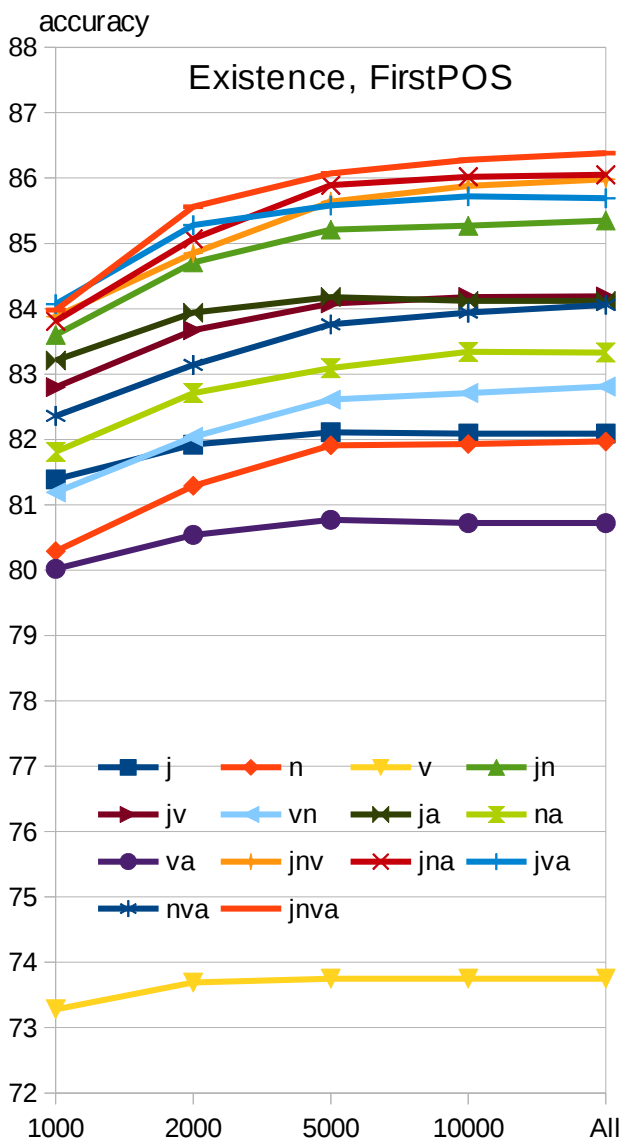


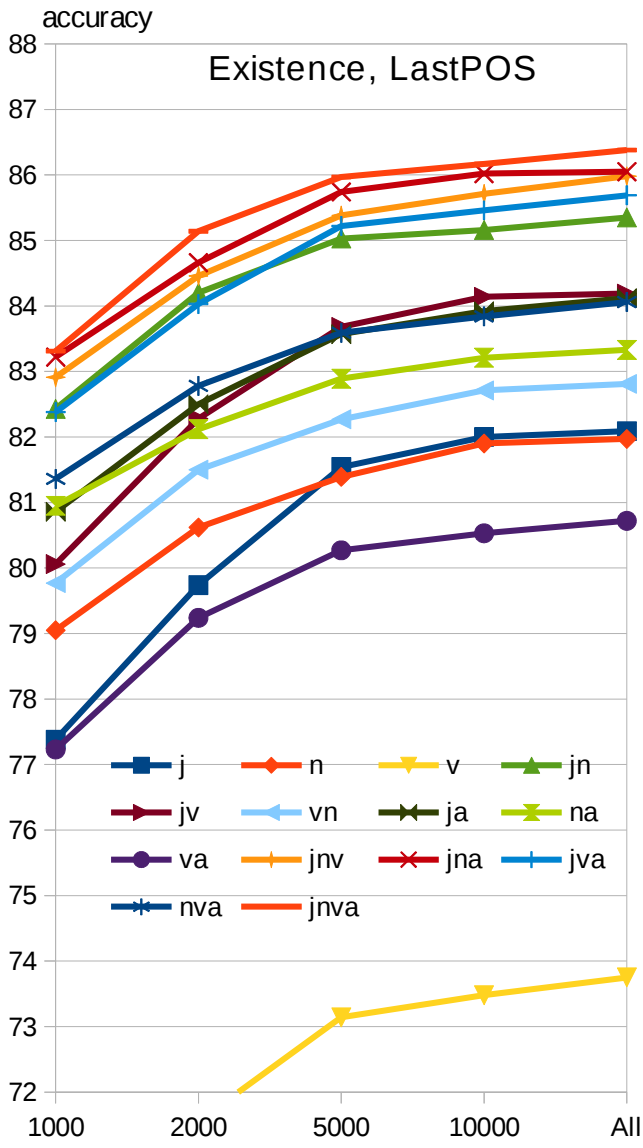
Final deductions from figure above are that considering existence over frequency, more words over less words, and FirstPOS over LastPOS will be likely to get higher results.


88
87
86
85
84
83
82
81
80
79
78
77
76
75
v
n
j
va
vn
na
ja
jv
jn
jnv
jna
jva
nva
jnva
Ex 1000 F
Ex 1000 L
Ex 2000 F
Ex 2000 L
Ex 5000 F
Ex 5000 L
Ex 10000 F
Ex 10000 L
Ex All F
Ex All L
Fr 1000 F
Fr 1000 L
Fr 2000 F
Fr 2000 L
Fr 5000 F
Fr 5000 L
Fr 10000 F
Fr 10000 L
Fr All F
Fr All L

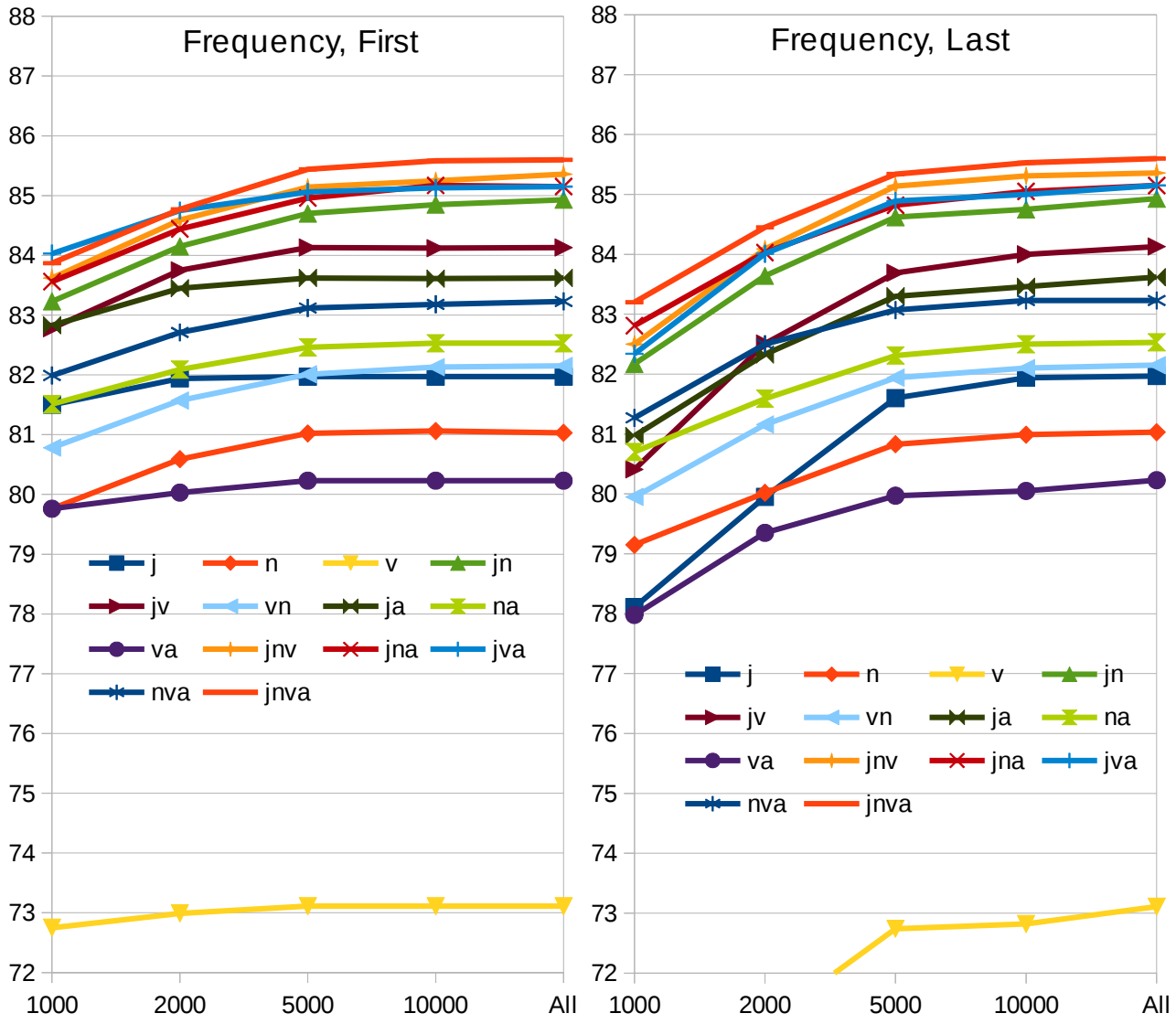
On the other hand, I have examined the effect of POS combinations. For the previous figure, I have shown accuracy of every possible combination except adverb alone as a POS combination. Abbreviations are Ex:Existence, Fr: Frequency, F: FirstPOS, L:LastPOS, ‘a’: adverb, ‘j’: adjective, ‘n’: noun, ‘v’: verb, for example; ‘ja’: ‘j’ and ‘a’.

As in the figure, adjective is the most important POS for classification by orientation of a document. Least important one is verb. The order goes like adj>noun>adverb>verb. The combination ‘ja’ or ‘jv’ is more effective than nva. This is more clearly indicated in next figures. We can get highest accuracy on ‘jnva’ when words’ existence is taken as input.



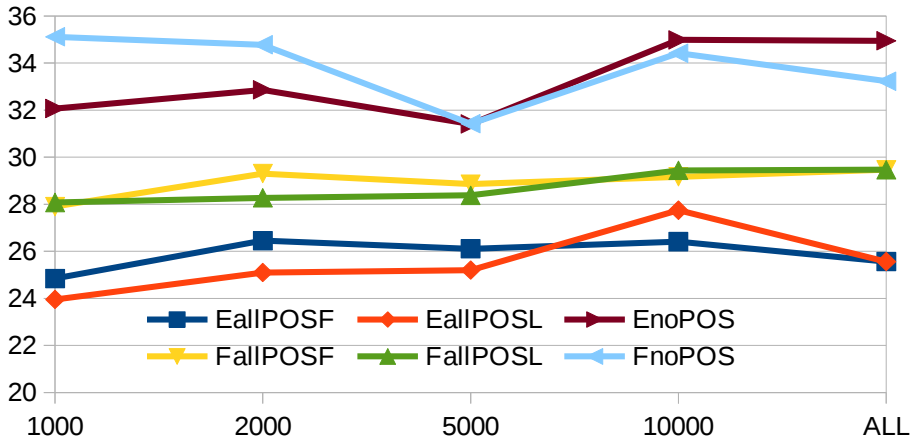


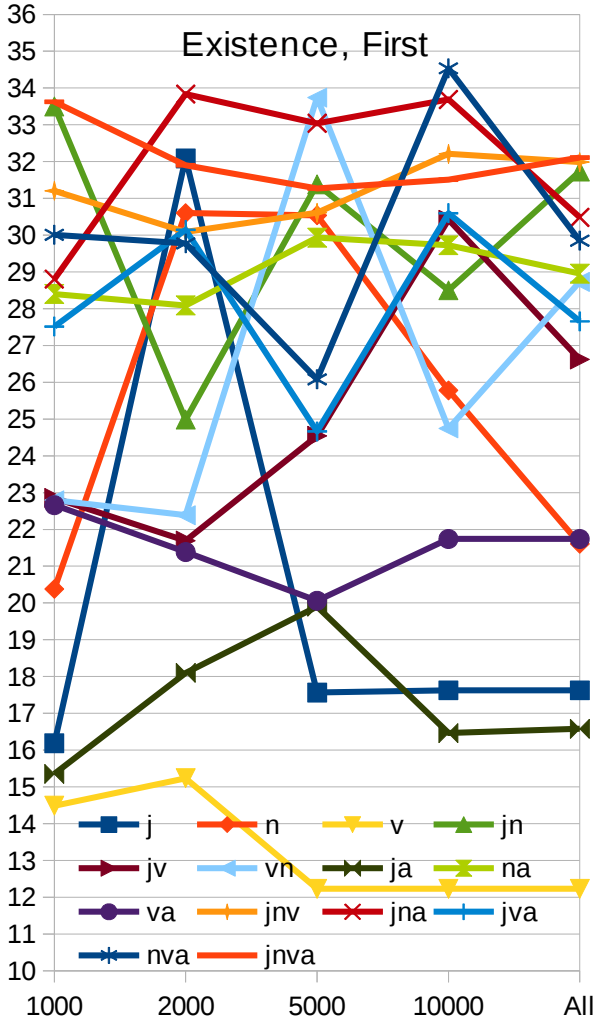
Previous deductions are also valid if frequencies of words are considered.

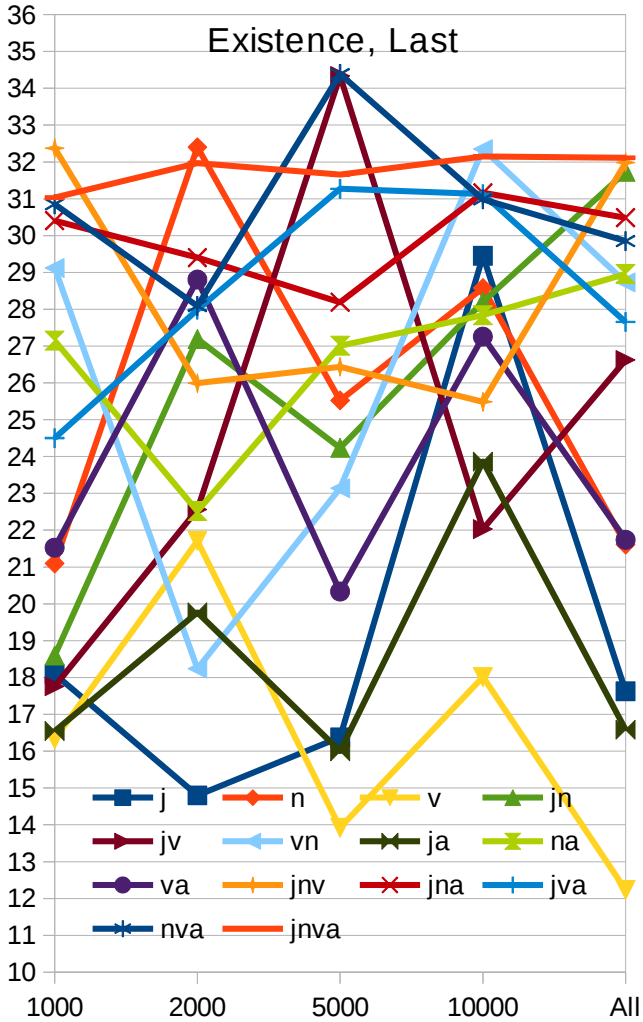


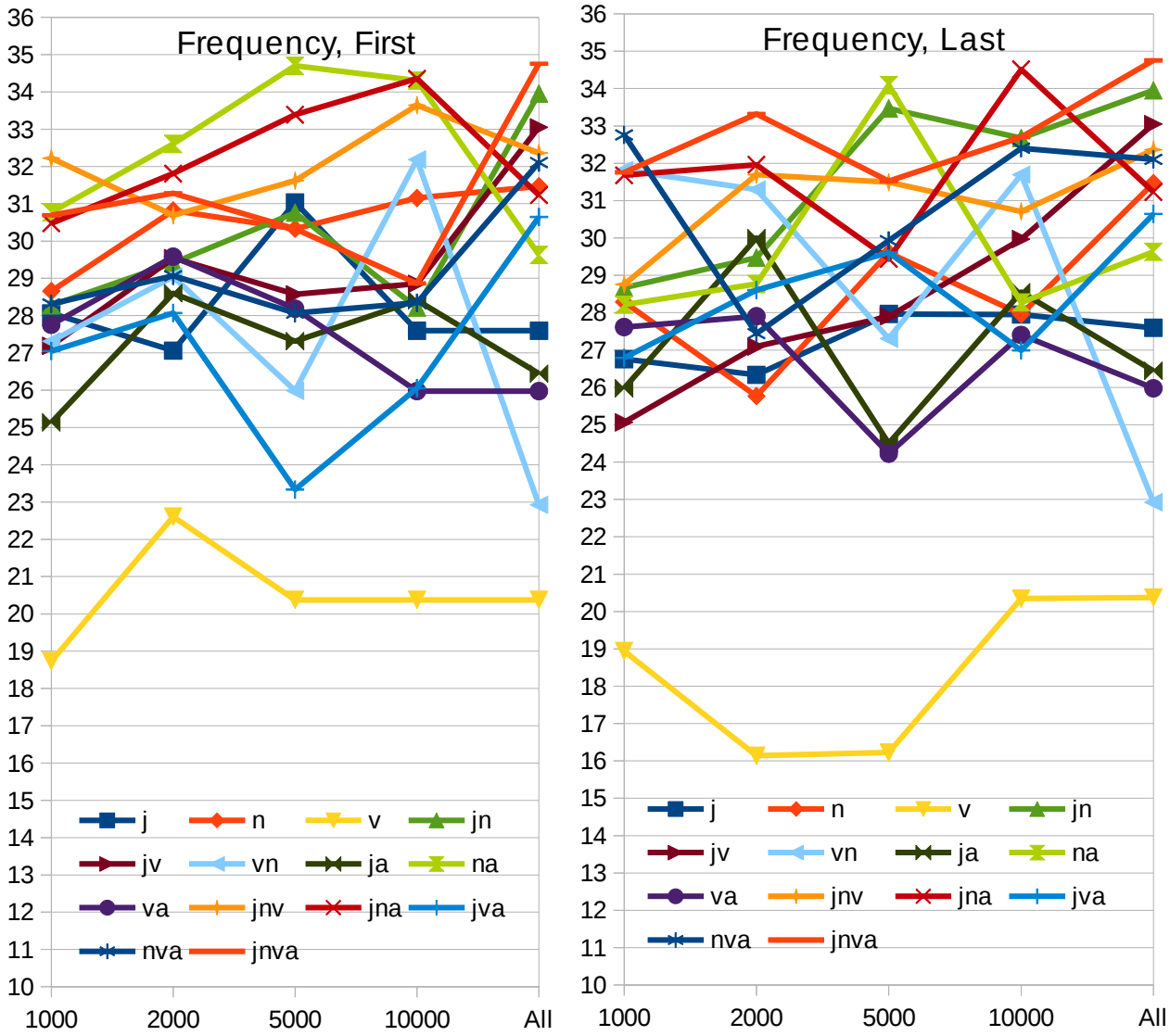
For sentiment analysis, if SVM is used for classification, number of features and accuracy usually change in similar manner. Previous figures can be probative for this claim. In figures titled as FirstPOS, accuracy of some combination of POSes do not change when number of words is increased. The reason is that total number of words with that POS combination is less than chosen number of words. For example, in IMDb vocab, considering ‘jv’, number of all adjectives plus number of all verbs is less than 10000.

For the other side of my project, rate prediction, I could reach very poor accuracy results which range between 15% and 40%. Like in sentiment analysis, I tried every POS combination. But it is almost impossible to make deductions from the low accuracy rates with high variance. In order to show what I meant, please look at following figures.









There are 8 classes (1,2,3,4,7,8,9,10) for rating in the dataset. Number of reviews for each rate is listed in following table:

Rate	1	2	3	4	7	8	9	10
#	10122	4586	4961	5331	4803	5859	4607	9731

If I classify them randomly, I could get 12,5% of accuracy. If I classify them as 1, I could reach 20.2% accuracy. But my rate prediction program can increase the accuracy up to 35% with POS combination as 'jnva'. Without dictionary(no POS), 35,5% of accuracy was achieved. Besides, when number of words increase, the accuracy does not necessarily increase. As a result, these results do not seem reliable.

To estimate the performance of my approach, I have used cross validation technique, specifically 5-fold cross validation. 50.000 reviews are firstly shuffled, and divided to 5 equal parts with 10.000 reviews. I created and named training data files as ‘Train0’, ‘Train1’, ‘Train2’, ‘Train3’, ‘Train4’. Each of files contains 40.000 reviews(four of 5 equal parts). Similarly, Test data files were created with same approach with name ‘Test’ rather than ‘Train’. Each of them contains 10.000 reviews(remaining one of 5 equal parts).

For sentiment analysis, I run SVM^{light} with following command:

```
$./svm_learn Train0 Model0
```

where “svm_learn” is SVM^{light} ‘s binary file for learning the each entries of training data, “Train0” is the training data file. “Model0” is the name of the file created after the command that includes all support vectors obtained from the learning process.

```
$./svm_classify Test0 Model0 Predict0
```

where “svm_classify” is SVM^{light} ‘s binary file for predicting the class of each entries of test data, “Test0” is the test data file. “Model0” is extracted from previous command. “Predict0” is the file that contains the predictions of each entry in the test data.
At the end of the output of the previous command, the accuracy rate is printed.

For rate prediction, I run SVM^{multiclass} with following command:

```
$./svm_multiclass_learn -c 1.0 Train0 Model0
```

where “svm_multiclass_learn” is SVM^{multiclass} ‘s binary file for learning.

```
$./svm_multiclass_classify Test0 Model0 Predict0
```

where “svm_muşticlass_classify” is SVM^{light} ‘s binary file for predicting.
At the end of the output of the previous command, the error rate is printed.

Discussion

As stated above, SVMs have been highly effective at sentiment classification. I have already shown it with my project with accuracy rate up to 86%, and I included other researches that have already shown it in this report. On the other hand, for rate prediction, it looks like SVM fails to classify correctly. I knew SVM is very poor at learning when it comes to multi-class classification. With this project, I had the chance to test this.
In terms of processing time, learning takes 2 minutes and predicting takes 5 seconds. Trying all combinations(all options and all POS combinations) takes approximately 18 hours. It can be easily said that sentiment classification requires a high computational power.
Besides, feature search and selection are formidable tasks. Representing the natural language in a simpler way is not necessarily an achievable mission.

Future Work

- Accuracy rates can be increased with additional features mentioned in State of Art section.
- SVM can be integrated with other algorithms that perform well on NLP.
- Some other datasets that are not related with movies should be used. This helps the program not to be dependent to a domain(movie), in other words, makes the program work on cross domain.
- Other languages can be considered to see how effective the approach is.

Conclusion

In the project, I have tried to make document-level sentiment analysis. For this purpose, I choosed SVM which is better at NLP. I found IMDb and modified it to be compatible with SVM classifiers(SVM^{light} and $SVM^{multiclass}$). I used Moby dataset as a dictionary for POS tagging of words in IMDb dataset. With dictionary, 86% accuracy is achieved. Without dictionary 87.3% accuracy is achieved at most. This seems to be very high compared to results of related works mentioned before. For rate prediction, the results are not very reliable to make a deduction.

References

[1] Sentiment Analysis (n.d.). In Wikipedia. Retrieved March 26, 2017, from http://en.wikipedia.org/wiki/Sentiment_analysis

[2]Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. "Thumbs up?: sentiment classification using machine learning techniques". In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10* (EMNLP '02), Vol. 10. Association for Computational Linguistics, Stroudsburg, PA, USA, 79-86.

[3] Tony Mullen, Nigel Collier, "Sentiment Analysis using Support Vector Machines with Diverse Information Sources" *Proceedings of EMNLP 2004*, pages 412–418, Barcelona, Spain. Association for Computational Linguistics

[4] Large Movie Review Dataset Retrieved May 5, 2017, from http://ai.stanford.edu/~amaas/data/sentiment/

[5] Moby POS Dataset Retrieved May 5, 2017, from http://icon.shef.ac.uk/Moby/mpos.html

[6] SVM^{light} Binary Files Retrieved May 5, 2017, from https://www.cs.cornell.edu/people/tj/svm_light/

[7] SVM^{multiclass} Binary Files Retrieved May 12, 2017, from https://www.cs.cornell.edu/people/tj/svm_light/svm_multiclass.html

[8] Liu, B., & Zhang, L. (2013). "A survey of opinion mining and sentiment analysis". In *Mining Text Data* (pp. 415-463). Springer US