Sentence Classification on Hebrew Texts According to Polarity

Tzeviya S. Fuchs[1], Dror Mughaz[1,2]

[1] Dept. of Computer Science, Lev Academic Center, 91160 Jerusalem, Israel
[2] Dept. of Computer Science, Bar-Ilan University, 52900 Ramat-Gan, Israel

tzevia@jct.ac.il, myghaz@gmail.com

EXTEND ABSTRACT

Sentiment analysis is considered to be a sub field of data mining, knowledge discovery and more specifically, of opinion mining. It deals with classifying written texts according to their polarity, i.e. as being positive or negative.

Automatic sentiment classification is important for marketing research, and is used mainly on data generated by internet users. Sentiment Analysis could be useful for companies that wish to find out what the world thinks of their products; for monitoring forums; or for analysis of customer feedback.

Some research has been conducted in this topic, but mostly for Latin languages, and no research has been done for Hebrew. This is important because the task of text classification is language-dependent. Hebrew, as a Semitic language, is considered to be a much tougher language than others, it being a highly inflected language with many ambiguous words. Therefore, classification methods that work well for English will not necessarily work so well when dealing with Hebrew.

Sentiment analysis is usually performed on one of the two levels: document level and sentence level. These two tasks are generally approached differently; in the document level there tend to be many unnecessary features, causing the need of aggressive feature selection. On the other hand, in the sentence level, it is common to try and "squeeze out" as much information possible from the features, as there are less of them available. The work on sentiment analysis for English texts was mostly performed on relatively long documents. Much less attention has been given to classifying very short text segments, like sentences. This is surprising; with the rising use of talkbacks and tweets generated by internet users, classifying short (and very informal) text segments becomes extremely important.

In this work, we focus specifically on classifying Modern Hebrew texts according to their polarity. Our corpus consists of short product reviews which were parsed into individual sentences.

Our baseline experiments are as follows: we constructed a negative-words list and a positive-words list, composed of the synonyms of the Hebrew words '*good*', '*excellent*', '*bad*' and '*very bad*'. The lists were used for a simple decision process that counts the number of positive and negative word that appear in the given texts, and classifies the text as belonging to the class to which most of its words belong. The experiment has been performed twice: once with the sentences in their raw form, and once after they have been lemmatized[1]. The results are as follows:

	Classified Correctly	**Classified Incorrectly**	**Unclassified**
Raw Form	24.73%	10.49%	64.78%
Lemmatized	34.75%	12.85%	52.40%

We then discuss the differences in the performances, the advantages and disadvantages of lemmatization, and how they affect the results.

In this work, we analyze various classification methods and their results on the corpus; we elaborate on the differences in classifying short texts versus long ones and about the uniqueness of working specifically with Hebrew; that is, how the morphology and even culture of Hebrew writers influence the results of our classification, and brought out our unique results.
In order to find an optimal categorization technique, we divided the classification process into three separate phases and studied each of them thoroughly. We put a particular emphasis on representational issues of feature extraction, as classification is language-dependent, and therefore the proper choice of features holds great importance.

The three phases are as follows:

1. *Choosing the feature vector:* previous research shows that the types of features chosen for the feature vector have a strong influence on classification accuracy. Various word representation-units are tested, such as using lemmas, negation tags, POS tags, unigrams, bigrams, stop-lists, etc. and are compared with results in English. In addition to these well-known methods, which had to be adjusted specifically for the special limitations of Hebrew, we added several optimizations and methods that have not been previously used, and could be applied for English as well. We analyze the results of the

[1] Lemma – the dictionary form of a word. In Hebrew it is usually the third person masculine *qal* perfect.

various feature vectors, and suggest upon some of the characteristics of Hebrew sentiment classification, and what makes it unique and different than other classification tasks.

The main results, evaluated with SVM, are as follows:

Firstly, using the most basic representation of a sentence (binary feature vectors of raw unigrams) yielded almost 80% accuracy, which obviously surpasses the baseline.

Negation tags proved to be effective, increasing performances in approximately 2%, which is quite unusual; negation tags are commonly used in English but without much effect on classification. The difference in the performances could be a result of working with a Hebrew corpus; it shows the different language structure of Hebrew, and perhaps shows a cultural difference between Hebrew writers to the writers of Latin languages.

Adding the negation tags was a little more complicated than it is in English; every Hebrew word has numerous inflected forms, and the negation word is not always explicit. Therefore, in addition to identifying negative words by a pre-defined negation word list, we also had to check the polarity of given words using a POS tagger.

Representing words by their lemmas yielded accuracies higher in 2% than when represented with their raw terms, and it obviously reduced the feature vector significantly. This, too, could in fact be a side effect of working with Modern Hebrew – the inflected words, the various spelling forms of them, and etc. could all be causes of unnecessary features (that lemmatization reduces).

One of the common problems in sentiment analysis is the presence of “thwarted expectations” sentences, that is, sentences that build up a certain impression and then conclude with a contradicting phrase. Therefore, we tried a simple approach that, upon detecting the word '*but*' (or one of its synonyms), removes the words that appear before it, assuming that they are irrelevant since the word '*but*' contradicted them. There was a slight, but negligible increase in performance.

Stop-word removal is commonly applied before classification. In our case, it had a negligible but slightly harmful effect on predictions. Out of a list of around 60 stop words, more than half turned out to be quite significant. Among the highest ranking stop words are ‘*and therefore*’, ‘*if*’ and ‘*but*’, which puts into question the liability of a manually constructed stop-list.

It should be noted that stop-word removal is Hebrew is slightly different and perhaps less necessary than it is in English, because typical English stop words such as ‘*a*’, ‘*and*’

or '*the*' either do not exist in Hebrew or are represented by prefixes (and are not removed with stop-lists).

Adding POS tags did not have a crucial impact on classification. With all of the ambiguity problems that exist in Hebrew, it would have seemed that adding POS tags is a vital. However, the light impact it had could be explained by the corpus being domain-specific, thus somewhat reducing ambiguity concerns. Another explanation is the inaccuracy of the POS tagger.

Further expanding the usage of POS tags, we experimented on finding whether any specific POS patterns that could indicate on sentiment repeated themselves throughout the corpus. Technically, we extracted bigrams and their POS tags, discarding the words to which the tags belonged, for the feature vector. In our case, accuracy rates are slightly lower, at 68%, and at 69%. However, the classifier did in fact find some interesting POS patterns, that indicate on very specific ways in which the writers of the texts express themselves, and show that not only the content of the texts are important, but also the way in which they were written.

2. *Applying feature selection:* there are various forms of feature selection, although they are not usually used on short text segments. We tested several methods, and proposed a new one, yielding surprisingly improved results.

   One commonly used form of feature selection involves the removal of infrequent words from the feature vector. The method turned out to be slightly harmful; some of the infrequent words that have been removed were assigned heavy weights by SVM as they were part of relatively short sentences.

   We then applied feature selection according to the weights of the features: when removing the features ranked less than 0.1 (by the SVM), performances jumped from 79.9% without feature selection to 86.5% with it. This contradicts previous works that show that SVM is not very sensitive to feature selection. The reason for the difference could be due to the amount of noise that exists in the corpus, as previously mentioned.

   We then propose a different method of feature selection: the object of the sentence has been chosen manually, and we then set different window sizes that determine how many of the words surrounding the aforementioned object should remain in the sentence. The rest of the sentence is discarded. Accuracies range from 62% with a window size of zero to 86% with a window size of five. Although not increasing performance, it provides an insight about what could perhaps be the relevant part of the sentence. This could be used

when there are concerns for the computational load of the corpus, or when sentences tend to be exceptionally long.

3. *Running the ML algorithm:* most of our experiments have been tested using SVM with a linear kernel; however, we compare SVM's results with those of Bayesian Logistic Regression and Voted Perceptron, which are variants of algorithms popularly used in text categorization.

   SVM yielded the highest results, as expected. However, Bayesian Logistic Regression provided surprisingly high results (but lower than SVM on average), that were many times very close to those of the SVM, and at times even slightly higher. Voted Perceptron yielded the worst results.

The best feature representations involved the combination of both unigrams and bigrams: it seems to be a natural choice, as spoken language is made of both individual words and phrases (phrases that are longer than 2 words are usually abbreviated when written). They were used with lemmas, with POS tags and without (with binary vector and weight feature selection), yielding 91% and nearly 92% accuracy (respectively). It should be noted that in these two cases, Bayesian Logistic Regression gave slightly higher results, 92.6% and 92.4% (respectively) which are exceptionally high. Previous somewhat similar works achieved a maximum of 85%-90%.