Feature enrichment through multi-gram models

Thomas Forss
Åbo Akademi
thomas.forss@abo.fi

Abstract

We introduce a feature enrichment approach, by developing multi-gram cosine similarity classification models. Our approach combines cosine similarity features of different N-gram word models, and unsupervised sentiment features, into models with a richer feature set than any of the approaches alone can provide. We test the classification models using different machine learning algorithms on categories of hateful and violent web content, and show that our multi-gram models give across-the-board performance improvements, for all categories tested, compared to combinations of baseline unigram, N-gram, and sentiment classification models. Our multi-gram models perform significantly better on highly imbalanced sets than the comparison methods, while this enrichment approach leaves room for further improvements, by adding instead of exhausting optimization options.

1. Introduction

Text classifications have developed in many different directions simultaneously, with a wide array of ideas and methods leading the development down different paths. Dimensionality reduction has been one of the corner stones of classification tasks, as classification performance can generally be improved by removing noise, no matter what type of data we are classifying. It is important to understand that dimensionality reduction consists of both feature extraction and feature selection [1]. Feature extraction is used to reduce dimensions in the data, and by doing the reduction create a set of features. Feature selection is the process of reducing that feature set and to remove the features that contain most noise from the classification model, to improve classification performance [2].

While it is true that we often can improve the classification performance through features selection, the performance gains are limited, as reducing noise can only be done to a certain point. If we in text classifications do feature selection prematurely, we risk doing the equivalent of painting ourselves into a corner, because, if performance is not at an acceptable level once we have performed feature selection, we have exhausted that option, and are left with little room for further improvement.

Many approaches in text classifications, such as bag-of-words approaches [3], N-gram approaches [4], and sentiment analysis [5], use feature extraction to reduce dimensions to a limited number of features, in some cases as low as one feature per class label [6]. However, it's worth noting that there are also other text extraction approaches, such as [7], [8], where the features sets can be much larger. These approaches with more features have shown promise in classifying shorter texts.

For the text classifications that have a limited number of features that are extracted, feature selection is not essential, unlike in cases where features number in tens or hundreds of thousands. When features are that many, it is possible that the training fails to complete, or that the amount of noise greatly reduces performance.

However, if we have feature sets that don't require feature selection to function, we suggest that it can be beneficial to first try other options that can improve text classification performance, such as for example feature set enrichment, before performing feature selection. Here, feature enrichment is defined as looking at ways of increasing the number of features, for example, by combining different feature extraction methods into a larger feature set.

In this paper, we will show that enriching the feature sets can improve performance across-the-board, over models with unigram, N-gram, and sentiment features. Previous research has also shown that combining sentiment features with cosine similarity features of different N-grams can improve performance [6], and we will compare our multi-gram performance against these models.

Many text classification models have problems performing well on highly skewed data, which can be seen as big data problem. For this purpose, we developed the multi-gram text classifications, and we can show that they perform well both on balanced and imbalanced data, when classifying web content. The approach is based on enriching feature sets, by combining cosine similarity features of different order N-grams, with unsupervised sentiment features, and using majority ensembles of different machine learning algorithms. Our approach is relevant to the literature, because it shows that through multi-grams we can remove the shortcoming that higher N-gram models have, which is that they require longer texts to make accurate decisions, and at the same time multi-grams show performance grains across-the-board compared to N-gram and unigram models.

The paper is structured as follows: In section 2, we discuss the big data dataset used for testing. In section 3, we discuss the multi-gram classification models, the comparison models, and the machine learning algorithms used in the experiments. In section 4, we compare the performance of the new models against the benchmarks both for balanced datasets and imbalanced data. In section 5, we offer some concluding thoughts, and discuss future work.

2. Dataset

The dataset used in the experiments, consists of 79063 manually labelled web pages split into 20 categories, which is one of the largest labelled datasets that we have worked with, which also makes it a good choice to test big data methods on. The list of categories, and their sizes, can be seen in Table 1. The dataset was gathered by a company in the security industry, and shared with us for research purposes. By request, we have worked mainly on categories 8, 12, 13, and 17, which have been identified as problematic categories to classify. Those four categories contain violent, hateful, and racist web pages. From this point forward, we will refer to categories 8, 12, and 13 as the hate categories, and category 17 as the violence category. The categories were labelled using a single labelling system, which means that there is a possibility of overlap between categories, especially between categories that share many words and thus have a high semantic similarity, such as, for example category 4 ‘Cigars’ and category 5 ‘Cigarette’, as well as the different hate categories.

Each web page in the dataset, consists of content extracted from 31 different HTML tags, such as for example keywords, URL, meta-content, full page text content, paragraph content, title content, and other often used HTML tags. We have tested many different combinations of different HTML tags as input to the classifications, and have found that by combining different parts of tags, we can adjust weights content in the web pages. It can be done either by excluding parts of the content, or by adding content several times to the input data that is forwarded to the feature extraction. In other experiments [6], it was found that including all content and adding extra weight by including the keywords, URL, and meta-content, a second time has worked well.

3. Classification Models

3.1. Similarity and Sentiment Models

The baseline classification models for violence and hate content that we follow are based on the ones used in [6]. The models combine cosine similarity features with unsupervised sentiment features. The cosine similarity features are extracted through TF-IDF weighting content of single web pages and comparing those to the TF-IDF weights of the pages in an entire class or category. TF-IDF weighing is calculated by counting the term frequency $tf_{t,d}$ and the inverse document frequency idf_t and multiplying them, see equations (1) – (3). By combining term frequency and inverse document frequency, we consider words that appear in many texts less important, while words that appear several times in a limited amount of texts are considered more important. This helps us get rid of language words that add little to no value. The range of values assigned starts from 0 and has no theoretical upper bound, although, in practice the value range is

Category	Description	Labelled Pages
1	Adult	6801
2	Beer	5913
3	Casino and gambling	3651
4	Cigars	1939
5	Cigarette	3845
6	Cults	3282
7	Dating	4703
8	Jewish hate	3479
9	Prescription drugs	5397
10	Occult	5105
11	Marijuana	6042
12	Racism,white supremacy	400
13	Racism againt minorities	4667
14	Religion	5438
15	Sports betting	2820
16	Spirits and liquor	3671
17	Violence	1919
18	Unknown	3432
19	Vine	4095
20	Weapons	2464
Total		79063

Table 1. List of the 20 categories in the classification dataset. We test our methods on categories 8, 12, 13, and 17 using features extracted from all the categories.

typically 0 to 10. Low scoring words are seen as of little importance and high scoring words are considered important: [9]

$$tf_{t,d} = f_{t,d} / \sum_{t' \in d} f_{t',d} \tag{1}$$

$$idf_t = \log\left(\frac{N}{1 + n_t}\right) \tag{2}$$

$$tfidf_{t,d} = tf_{t,d} * idf_t \tag{3}$$

The term frequency and inverse document frequency calculations in (1) and (2), are applicable only for unigram extraction. If we want to do higher N-gram extraction, we need to define t as follows, where i is the N-gram number we are extracting, and W are the words included in each N-gram: [4]

$$t = t_i(W_i \dots W_{i+n}) \tag{4}$$

Cosine similarity $sim(d_i, c_j)$ is then calculated between the category c_j and the web page d_i as in equation (5), where d_i is the $tfidf_{t,d}$ vector extracted for the web page i using (3), and c_j is the $tfidf_{t,d}$ vector extracted for the category j also using (3). If we do the cosine similarity calculation for each category $j = \{1, \dots, 20\}$ in the dataset, we get a feature set y_{sim}, as represented in equation (6), containing 20 cosine similarity features, one feature for each class: [10]

$$sim(d_i, c_j) = \frac{d_i \cdot c_j}{\|d_i\| \|c_j\|} \tag{5}$$

$$y_{sim}(i) = \{sim(d_i, c_1), \dots, \mathrm{sim}(d_i, c_{20})\} \tag{6}$$

Unsupervised sentiment features are extracted from the texts using the dictionary based approach introduced by [11]. The approach extracts sentiment values for words on a scale of -5 to +5, by using dictionaries containing English words manually labelled as positive (+1 to +5) or negative (-1 to -5). Through using an extension of the sentiment extraction algorithm [6, 12], we extract 13 sentiment features for each text. In table 2, the sentiment features are described in more detail, and in equation (7), we represent the sentiment feature set y_{sent} mathematically.

$$y_{sent} = \{y_1, \dots, y_{13}\} \tag{7}$$

When these two sets of similarity features and sentiment features are added together, we call the aggregate with 33 features y_{tot}, as shown in equation (8), where n denotes the N-gram used. This type of feature extraction will be used as the baseline classification comparisons. The baseline will be using unigram, one-gram, tri-gram, and five-gram features. The unigram and one-gram approaches both follow the single-word TF-IDF calculations $y_{sim(1)}$ from equation (5), however, the IDF calculations of the unigram and one-gram approaches are calculated differently. Our unigram calculations use an existing IDF-dictionary developed by [13], while the rest of our IDF-calculations are calculated based on our own dataset. That gives us the unique possibility of comparing performance between our own IDF-calculations and the IDF-calculations extracted from a more generalized dataset.

$$y_{tot(n)} = \{y_{sim(n)}, y_{sent}\} \tag{8}$$

3.2. Multi-gram Models

Two general problems have been identified when using N-gram classification models, and they are likely to affect the models in equation (8) that combine similarity with sentiment. The first problem is that N-gram models have varied classification performance when applied to different length texts, and is something that has been previously discussed in the literature. Cavnar et al. [4] found that N-grams perform slightly better on longer texts. The main reason for that is attributed to N-gram models having few or sometimes even no IDF matches on shorter texts. Liu et al. [14] surveyed different methods used to improve performance on short texts, many of them based on N-gram models. Examples of situations when this becomes important is classifications of news headlines or tweets. In both those cases there is a risk that the texts being classified will contain only a few words, which leads to just a few or even no N-gram matches. Furthermore, the classification performance is simply lacklustre in many cases when using unigrams, no matter what combination of machine learning algorithms used. In this research, we will see an example of insufficient performance

$y_1 - y_5$	Sum of negative words, grouped by strength [-1, -2, -3, -4, -5]
$y_6 - y_{10}$	Sum of positive words, grouped by strength [1, 2, 3, 4, 5]
y_{11}	The most positive and the most negative word in the text
y_{12}	Sum of all sentiment words values in the text
y_{13}	Number of positive sentiment words minus number of negative sentiment words

Table 2. Description of sentiment features

when using unigram extraction on violence and hate content.

The second problem, is a general classification problem, which becomes evident only when we move classifications from a balanced test setting into something that can better represent real life big data skews. In binary classifications, there are very few, if any, real-world scenarios where the data is evenly split between the positive and the negative category. The result of testing classifiers on highly skewed data that were created using balanced datasets, is generally the same: The performance drops for all threshold measures [15].

Training models from the start on greatly imbalanced datasets doesn't solve the problem either, but it does increase the time it takes to train the models. The approaches generally used to combat the drops in performance, are either under sampling the class with more instances when training, or oversampling the class with fewer instances [16]. Other approaches to improving performance on imbalanced sets, are using cost sensitive learning methods [17] and ensemble learning [18]. Through cost sensitive learning it is possible to define that misclassifying one class is more costly than misclassifying the other [17]. Through ensemble learning it is possible to combine several models to improve the results [18]. Support Vector Machines (SVM) can be used to change the costs of misclassifications [19]. Still, there is one lingering question: What do we do in situations where none of these techniques presented are enough to give acceptable performance?

To add one more option to solve such problems, we introduce the multi-gram classification model. The model has some similarities to the methods used by [20], [21]. However, in our multi-gram model, we extract features from all N-grams up until the chosen max n, creating the possibility of using many features for each class, where in [20], they try to determine the N-gram feature that is most suitable, through a method that they call n-multigram extraction. The generalized multi-gram feature set y_{mg} that we use, can be represented as in equation (9), where n denotes the highest N-gram for which features are extracted:

$$y_{mg} = \{y_{sim(1)}, \dots, y_{sim(n)}, y_{sent}\} \quad (9)$$

In our multi-gram tests, we use cosine similarity feature extraction for unigram, one-gram, tri-gram, and five-gram features. We could also have included two-grams and four-grams, but chose not to, due to time constraints. More comparisons between multi-grams will be done as part of future work. We chose to include five-grams instead of a lower N-gram, for the reasons that five-gram models were also included in the comparison models [6], and that other studies have shown that N-gram models with higher than tri-grams tend to not improve classification results [22]. We want to find out whether this is also the case with multi-gram classification models. We test two different multi-gram models, one using a set of 80 cosine similarity features represented in equation (10), and a second model that adds the 13 sentiment features that were discussed earlier, shown in equation (11), allowing the second model to use a total of 93 features:

$$y_{mg_1} = \{y_{sim(uni)}, y_{sim(1)}, y_{sim(3)}, y_{sim(5)}\} \quad (10)$$

$$y_{mg_2} = \{y_{sim(uni)}, y_{sim(1)}, y_{sim(3)}, y_{sim(5)}, y_{sent}\} \quad (11)$$

3.3. Machine Learning Algorithms

In the performance tests, we will be comparing the performance of four machine learning algorithms: Decision Trees (DT) [23], k-Nearest Neighbors (k-NN) [24], Support Vector Machines (SVM) [25], and Artificial Neural Networks (ANN) [26], to see how the benchmark models compare to the multi-gram models using the different machine learning algorithms. Furthermore, we will also test how a majority voting ensemble classification of those four algorithms impacts the performance [18]. Ensemble algorithms have been shown to improve performance on imbalanced data [16]. Finally, we will extend the tests from balanced testing to imbalanced testing with a data skew of roughly 20 as defined in [15], which means that in the final tests, the models with be tested using roughly 5% positive instances and 95% negative instances. That should give us an understanding of how the multi-gram models perform both on balanced and imbalanced data.

4. Results

We start by testing the baseline performance of the models using the four machine learning algorithms (DT, k-NN, SVM, ANN), on the four categories in the dataset that was discussed in section 2, using ten-fold cross validation. The categories tested are numbered 8, 12, 13, and 17 in Table 1. The tests are performed using 33 features where the similarity features change but the sentiment features stay the same, as they are not affected by changes in N-gram size. In Table 3, we can see the performance results for the classifiers using standard and automatic settings for the different algorithms using RapidMiner for classifications [27].

Automatic settings mean that, for instance, the neural networks themselves determines the number of hidden layers, which then change depending on the categories being classified and the features used. The unigram and one-gram performance is quite close in performance, even though we used different IDF calculation methods. Because of that, we choose to only include the unigram results in Table 3. We can also see that the ANN and SVM performance is higher than the other algorithms on average, both algorithms have been shown to have good performance in text classification tasks. Our Tri-gram and five-gram results are respectable using the ANN and balanced test sets. Micro F-Measures above 0.90 for all four classes using ANN.

The performance of these models looks promising when testing them on balanced data, however, we will later see that the performance drops significantly on imbalanced data to the point that these models wouldn't be useful.

We continue the baseline testing by doing combinations of majority voting ensembles. We test three different ensembles: SVM/ANN, DT/SVM/ANN, and DT/SVM/ANN/k-NN. The ensemble of all algorithms is labelled as "All" in the result tables, and the ensemble of DT/SVM/ANN is not included as it has the worst over-all performance. We can also see the results of the ensembles on balanced data in Table 3. The only performance improvement that ensemble models offer over previous runs using ANN models, was for category 12, using tri-grams and the DT/SVM/ANN majority ensemble. For the other categories, the ANNs showed the best performance. The ensembles do reduce model variance on average in the classifications, but fail to deliver overall performance increases on the balanced datasets. The most likely reason that ANNs work well on this type of data is that they thrive in cases where there is much data.

4.1. Multi-gram Model Testing

As we now have defined the performance of the benchmarks, we move on to the multi-gram model performance on balanced datasets. Here, we limit our experiments to the best performing models, which are the different ensemble models, SVM, and ANN, and we will only look at performance at a threshold of 0.5. The results for the multi-gram experiments can be seen in Table 4. Comparing the multi-gram models against the benchmarks, we can see that both the multi-gram models including sentiment features and the multi-gram models not including them show higher accuracy and F-Measures in each category.

Comparing balanced set performance between the two different multi-gram models, we can see that for some categories, the best performance is achieved when including sentiment features (8, 13, and 17), however, the margin is quite small and more testing will be needed before we can draw generalizing conclusions. For category 12, the best performance was found when not including sentiment features. Furthermore, we can see that the variance in model performance also varies between the different categories and models. In other words, including sentiment features increase multi-gram performance in some cases, and in some other situations they don't.

Comparing different algorithms, we can also see that ANNs still perform best – alone – on balanced data, with some of the other methods having higher precision, recall, and/or lower model variance, but not better overall performance.

4.2. Imbalanced Dataset Testing

One of the problems that we identified earlier, was that going from balanced sets to skewed data reduces performance. To simulate real-world data skews, we will now look at the performance of the algorithms at a skew close to 20 for the test sets, while still using a balanced set for training, an approach that is known as under sampling, reducing the negative class size when training [16]. However, when doing these tests, we will no longer be performing cross-validation on the experiments, as we don't want to reduce the number of positive testing instances further. That means the results are less generalizable, and more susceptible to outliers, but are still comparable between the models used, as long as we use the same test sets when comparing categories using different models.

Furthermore, as the data is skewed, the accuracy, precision, and recall measures will be misguiding. Instead, we will here look at the F-Measure values when comparing the threshold performance of the models [15]. In Table 5, to the left, we can see the tri-gram performance using imbalanced testing data. The F-Measure performance drops significantly using tri-gram models with imbalanced data. In fact, the performance in many cases drops below the threshold where we would get better accuracy by simply classifying everything as negative. Only category 13 and 17 show somewhat useful results, both show best performance using the majority ensemble of SVM/ANN.

In Table 5, to the right, we can see the performance of the multi-gram models using the same imbalanced data with different machine learning algorithms. Category 8 has the best imbalanced performance using ANN. Category 12 has the best performance using the ensemble of SVM/ANN, and comes in at the best imbalanced result with an F-Measure of 0.83. Category 13 has the lowest performance of the four categories, also using ensemble of SVM/ANN without sentiment features. An ensemble of all models is the best performing model for category 17.

Using this specific imbalance in data, the models without sentiment features showed slightly better performance. However, when we tested datasets with slightly smaller imbalance, the models with sentiment features performed better overall. The conclusion that we can draw based on these results is that our multi-gram approach performs better on imbalanced data than the benchmark methods.

Here, we would still have plenty of options to choose from, if we want to continue maximizing the performance of the models, as the approach so far has not exhausted any of the standard performance increasing techniques. We could now, for example, choose to perform feature selection or testing different machine learning algorithm hyper-parameters, or alternatively, we could further enrich the feature sets with more multi-grams, before applying the other performance improving techniques.

Finally, in Table 6, we take a closer look at the false positives of the best performing category model 12, which had the F-Measure of 0.83. We can see that there are 16 false positives in total, and a large overlap between the hate categories 12 and 13, where 13 of the total 16 false positives are labelled as category 13. In a practical application, we would most likely not consider labelling one hate category as another hate category as a misclassification – they would be treated the same way. Thus, if we exclude those category 13 false positives from the performance calculations, we produce slightly relaxed labelling conditions, and can re-calculate the "practical" performance for category 12. The F-Measure we get with the relaxed conditions comes in at an impressive 0.93 for the imbalanced data using category 12.

If we are interested in comparing the imbalanced results to other classification areas, we can look at the results from the web spam challenges, where researchers have classified web spam datasets with the same skew between positive and negative instances [28]. While the results cannot be directly compared without running our models on the same data, we can still do a rough comparison, as context in both scenarios are web pages and the set imbalance is similar.

However, the results in those challenges are reported using the performance measure 'area under the receiver operating characteristic' (AUROC), which we need to compute to be able to compare the results. The best performing models that [28] presented on spam filtering, has showed average AUROC of 0.892. Our multi-gram models using the best performing majority ensemble of NN/SVM, have an average AUROC of 0.950 for multi-grams without sentiment features and 0.952 for multi-grams with sentiment features. The AUROC measure, can be read the following way: A measure of 0.8 to 0.9 is considered good performance, a measure of 0.9 to 1 is considered excellent [29]. The multi-gram models seem to be competitive, however, to know if they can be used also in other cases, we need to test the using other types of data.

5. Conclusion

We introduced a new text classification model, the multi-gram model that combines features of different N-grams with sentiment features, which showed an across-the-board performance increase over the comparison methods. We were able to achieve F-Measures of 0.93 and above on balanced datasets using the models, and found that the majority ensemble didn't improve the threshold measure performance above artificial neural networks on balanced test data. However, when we tested the same models on highly skewed dataset, with distributions close to 95% negative instances, we found that in most situations the models without ensembles dropped significantly in performance.

Our multi-gram models were able to show good performance, compared to the other benchmark models that had higher performance drops for the imbalanced data tests. In many cases, the performance drops using the best performing benchmark models, dropped the accuracy down to below the data skew split, meaning that placing all instances in the negative category would have shown better accuracy results. Our multi-gram models, on the other hand, performed well using all measures, and had out-of-the box F-Measures ranging from 0.71 to 0.83 for all four categories using the imbalanced test data.

5.1. Future Work

Our research will in the future be extended in several different directions. First, the research will be extended and performed on all 20 categories in the dataset to give a better understanding of how the models perform on different types of categories. For the same purpose, we will also extend the research and testing to the web spam data sets, to get direct comparisons to performance of other models.

To improve the models, we will test different multi-gram models and compare performance between them. We will also research whether feature selection and machine learning parameter optimization can further improve performance of the classification models.

Finally, we are also interested in extending the research to include features from other approaches that have shown potential on shorter texts, such as the approach taken by Kusner et al. [7].

Balanced performance												
Category 8	Unigram				Tri-gram				Five-gram			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
DT	79.84	0.82	0.77	0.79	88.69	0.91	0.86	0.88	89.19	0.85	0.95	0.90
SVM	83.64	0.83	0.85	0.84	89.09	0.92	0.86	0.89	89.98	0.96	0.83	0.89
ANN	87.12	0.87	0.87	0.87	92.85	0.94	0.91	0.93	92.38	0.96	0.89	0.92
k-NN	79.81	0.77	0.85	0.81	80.78	0.78	0.86	0.82	80.32	0.77	0.85	0.81
SVM/ANN	84.73	0.80	0.92	0.86	90.88	0.90	0.92	0.91	91.99	0.94	0.90	0.92
All	86.16	0.83	0.90	0.87	91.42	0.92	0.91	0.91	92.95	0.95	0.89	0.92
Category 12	Unigram				Tri-gram				Five-gram			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
DT	72.85	0.66	0.93	0.77	92.02	0.93	0.91	0.92	91.15	0.95	0.86	0.91
SVM	81.93	0.79	0.87	0.82	84.15	0.97	0.70	0.81	81.57	0.96	0.65	0.78
ANN	80.83	0.80	0.81	0.81	94.23	0.94	0.94	0.94	92.39	0.93	0.91	0.92
k-NN	66.34	0.64	0.71	0.67	74.81	0.74	0.75	0.75	73.10	0.71	0.76	0.73
SVM/ANN	80.71	0.83	0.76	0.79	84.52	0.98	0.86	0.88	82.31	0.98	0.65	0.78
All	81.93	0.80	0.84	0.82	91.52	0.99	0.88	0.88	89.80	0.98	0.81	0.89
Category 13	Unigram				Tri-gram				Five-gram			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
DT	73.29	0.69	0.84	0.76	86.38	0.87	0.86	0.86	85.20	0.94	0.75	0.84
SVM	79.25	0.79	0.80	0.79	84.13	0.94	0.73	0.82	83.37	0.96	0.69	0.81
ANN	83.88	0.83	0.86	0.84	90.60	0.92	0.89	0.90	90.46	0.92	0.89	0.90
k-NN	78.30	0.76	0.83	0.79	79.90	0.78	0.83	0.80	79.86	0.79	0.81	0.80
SVM/ANN	81.43	0.87	0.74	0.80	84.91	0.97	0.72	0.83	83.70	0.98	0.69	0.81
All	82.84	0.84	0.81	0.82	88.52	0.95	0.81	0.88	86.84	0.98	0.75	0.85
Category 17	Unigram				Tri-gram				Five-gram			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
DT	62.53	0.93	0.27	0.41	88.69	0.95	0.81	0.88	86.54	0.94	0.78	0.85
SVM	82.48	0.91	0.72	0.80	86.46	0.98	0.74	0.85	84.36	0.98	0.70	0.82
ANN	83.06	0.84	0.82	0.83	92.62	0.92	0.93	0.93	90.90	0.90	0.92	0.91
k-NN	70.69	0.69	0.74	0.72	77.03	0.76	0.79	0.77	75.68	0.76	0.74	0.75
SVM/ANN	82.17	0.93	0.69	0.79	86.59	0.99	0.74	0.85	82.23	0.99	0.69	0.81
All	77.68	0.94	0.59	0.72	88.85	0.98	0.79	0.88	86.41	0.98	0.75	0.85

Table 3. Unigram, tri-gram, and five-gram classification performance on balanced data.

Balanced multi-gram performance								
Category 8	Multi-gram 80 features				Multi-gram 93 features			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
SVM	92.59	0.96	0.89	0.92	92.64	0.96	0.89	0.92
ANN	94.18	0.96	0.93	0.94	94.48	0.94	0.96	0.95
SVM/ANN	94.01	0.94	0.94	0.94	94.19	0.93	0.96	0.94
All	94.29	0.95	0.93	0.94	94.02	0.94	0.94	0.94
Category 12	Multi-gram 80 features				Multi-gram 93 features			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
SVM	92.88	0.95	0.90	0.93	92.63	0.93	0.91	0.92
ANN	95.82	0.95	0.97	0.96	95.45	0.95	0.96	0.95
SVM/ANN	93.74	0.98	0.89	0.93	94.23	0.97	0.91	0.94
All	94.97	0.97	0.92	0.95	94.10	0.96	0.92	0.94
Category 13	Multi-gram 80 features				Multi-gram 93 features			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
SVM	89.92	0.93	0.85	0.89	89.80	0.93	0.86	0.89
ANN	92.39	0.92	0.93	0.92	92.45	0.92	0.93	0.93
SVM/ANN	90.34	0.96	0.84	0.90	90.78	0.96	0.85	0.90
All	91.78	0.96	0.87	0.91	90.77	0.96	0.85	0.90
Category 17	Multi-gram 80 features				Multi-gram 93 features			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
SVM	91.27	0.97	0.85	0.91	91.06	0.98	0.83	0.90
ANN	92.80	0.94	0.94	0.94	94.07	0.94	0.94	0.94
SVM/ANN	91.58	0.98	0.85	0.91	91.03	0.96	0.85	0.90
All	92.02	0.97	0.87	0.92	92.49	0.99	0.86	0.92

Table 4. Multi-gram classification performance on balanced data.

Imbalanced performance												
Category 8	Tri-gram				Multi-gram 80 features				Multi-gram 93 features			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
SVM	93.08	0.40	0.88	0.55	95.81	0.54	0.88	0.67	95.60	0.53	0.88	0.66
ANN	92.80	0.40	0.94	0.56	96.72	0.61	0.90	0.73	95.43	0.52	0.92	0.66
SVM/ANN	89.41	0.31	0.96	0.47	94.50	0.47	0.93	0.62	93.48	0.42	0.94	0.58
All	91.35	0.35	0.94	0.51	95.06	0.49	0.92	0.64	94.35	0.46	0.94	0.62
Category 12	Tri-gram				Multi-gram 80 features				Multi-gram 93 features			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
SVM	89.39	0.28	0.69	0.40	96.89	0.63	0.91	0.75	96.16	0.57	0.93	0.71
ANN	85.59	0.29	0.94	0.44	97.16	0.65	0.96	0.77	97.07	0.64	0.98	0.77
SVM/ANN	90.48	0.30	0.69	0.42	98.08	0.76	0.91	0.83	97.90	0.73	0.93	0.82
All	86.00	0.24	0.80	0.37	97.26	0.66	0.95	0.78	96.61	0.61	0.93	0.73
Category 13	Tri-gram				Mult0.66i-gram 80 features				Multi-gram 93 features			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
SVM	94.32	0.44	0.70	0.54	94.28	0.45	0.81	0.58	94.25	0.45	0.82	0.58
ANN	96.62	0.58	0.74	0.65	94.41	0.46	0.91	0.61	93.17	0.41	0.92	0.57
SVM/ANN	96.50	0.62	0.70	0.66	96.95	0.63	0.80	0.71	96.82	0.64	0.80	0.71
All	94.42	0.48	0.79	0.60	96.60	0.61	0.85	0.71	96.85	0.64	0.80	0.71
Category 17	Tri-gram				Multi-gram 80 features				Multi-gram 93 features			
Algorithm	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.	Acc.	Prec.	Recall	F-Meas.
SVM	96.51	0.60	0.73	0.66	95.94	0.54	0.86	0.66	96.85	0.62	0.83	0.71
ANN	89.28	0.27	0.95	0.43	94.64	0.46	0.95	0.62	79.70	0.18	0.99	0.31
SVM/ANN	96.86	0.64	0.73	0.68	97.38	0.67	0.85	0.75	96.95	0.63	0.83	0.72
All	96.58	0.60	0.75	0.67	97.40	0.67	0.87	0.76	96.57	0.59	0.86	0.70

Table 5. Imbalanced classification performance using the best performing models.

List of False Positive Classifications Category 12				
Category	16 False Positives:			
Casino (3)	1			
Racism (13)	13			
Sports betting (15)	2			
Re-calculated Imbalanced Performance Category 12				
Algorithm	Accuracy	Pre.	Recall	F-Me.
SVM/ANN	99.26%	0.94	0.91	0.93

Table 6. Relaxed classification conditions can improve performance when we don't consider it wrong to classify one hate category as another hate category.

6. Acknowledgements

I would like to thank Professor Dr. Christer Carlsson and Dr. Kaj-Mikael Björk for their help in my work and studies.

7. References

[1] D. D. Lewis, "Feature Selection and Feature Extraction for Text Categorization," in Proceedings of the Workshop on Speech and Natural Language, Stroudsburg, PA, USA, 1992, pp. 212–217.

[2] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," J Mach Learn Res, vol. 3, pp. 1157–1182, Mar. 2003.

[3] Y. Zhang, R. Jin, and Z.H. Zhou, "Understanding bag-of-words model: a statistical framework," International Journal of Machine Learning and Cybernetics, 1(1-4), pp.43-52, 2010.

[4] W. B. Cavnar, J. M. Trenkle, and others, "N-gram-based text categorization," Ann Arbor MI, vol. 48113, no. 2, pp. 161–175, 1994.

[5] B. Pang, L. Lee, and V. Shivakumar, "Thumbs up?: sentiment classification using machine learning techniques," In Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10, pp. 79-86, 2002.

[6] S. Liu and T. Forss, “Combining N-gram based Similarity Analysis with Sentiment Analysis in Web Content Classification.,” in KDIR, 2014, pp. 530–537.

[7] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, “From Word Embeddings To Document Distances,” in Proceedings of The 32nd International Conference on Machine Learning, 2015, pp. 957–966.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” ArXiv Prepr., 2013.

[9] H. P. Luhn, “The automatic creation of literature abstracts,” IBM J. Res. Dev., vol. 2, no. 2, pp. 159–165, 1958.

[10] Z. Markov and D. T. Larose, Data mining the Web: uncovering patterns in Web content, structure, and usage. John Wiley & Sons, 2007.

[11] M. Thelwall, K. Buckley, and G. Paltoglou, “Sentiment in Twitter events,” J. Am. Soc. Inf. Sci. Technol., vol. 62, no. 2, pp. 406–418, Feb. 2011.

[12] S. Liu and T. Forss, “Web Content Classification based on Topic and Sentiment Analysis of Text.,” in KDIR, 2014, pp. 300–307.

[13] D. R. Radev, H. Jing, M. Styś, and D. Tam, “Centroid-based summarization of multiple documents,” Inf. Process. Manag., vol. 40, no. 6, pp. 919–938, 2004.

[14] Z. Liu, W. Yu, W. Chen, S. Wang, and F. Wu, “Short text feature selection for micro-blog mining,” Computational Intelligence and Software Engineering, 2010.

[15] L. A. Jeni, J. F. Cohn, and F. De La Torre, “Facing imbalanced data–Recommendations for the use of performance metrics,” in Affective Computing and Intelligent Interaction (ACII), 2013, pp. 245–251.

[16] Y. Liu, H. T. Loh, and A. Sun, “Imbalanced text classification: A term weighting approach,” Expert Syst. Appl., vol. 36, no. 1, pp. 690–701, 2009.

[17] C. Elkan, “The foundations of cost-sensitive learning,” in International joint conference on artificial intelligence, 2001, vol. 17, pp. 973–978.

[18] T. G. Dietterich, “Ensemble methods in machine learning,” in International workshop on multiple classifier systems, 2000, pp. 1–15.

[19] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser, “SVMs modeling for highly imbalanced classification,” IEEE Trans. Syst. Man Cybern. Part B Cybern., vol. 39, no. 1, pp. 281–288, 2009.

[20] D. Shen, J.-T. Sun, Q. Yang, and Z. Chen, “Text classification improved through multigram models,” presented at the Proceedings of the 15th ACM international conference on Information and knowledge management, 2006, pp. 672–681.

[21] H. M. Wallach, “Topic modeling: beyond bag-of-words,” in Proceedings of the 23rd international conference on Machine learning, 2006, pp. 977–984.

[22] J. Fürnkranz, “A study using n-gram features for text categorization,” Austrian Res. Inst. Artifical Intell., vol. 3, no. 1998, pp. 1–10, 1998.

[23] S. J. Russell and P. Norvig, “Artificial Intelligence: A Modern Approach,” 2002.

[24] D. T. Larose, “k-Nearest Neighbor Algorithm,” Discov. Knowl. Data Introd. Data Min., pp. 90–106, 2005.

[25] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, “New support vector algorithms,” Neural Comput., vol. 12, no. 5, pp. 1207–1245, 2000.

[26] R. Collobert and S. Bengio, “Links between perceptrons, MLPs and SVMs,” in Proceedings of the twenty-first international conference on Machine learning, 2004, p. 23.

[27] “RapidMiner” [Online]. Available: https://rapidminer.com/ [Accessed: 21-Sep-2016].

[28] M. Erdélyi, A. Garzó, and A. A. Benczúr, “Web spam classification: a few features worth more,” in Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality, 2011, pp. 27–34.

[29] “The Area Under an ROC Curve.” [Online]. Available: http://gim.unmc.edu/dxtests/roc3.htm [Accessed: 16-May-2017].