Second Order Features for Maximising Text Classification Performance

Bhavani Raskutti, Herman Ferrá, and Adam Kowalczyk

Telstra Corporation, 770 Blackburn Road, Clayton, Victoria 3168, Australia
{Bhavani.Raskutti, Herman.Ferra, Adam.Kowalczyk}@team.telstra.com

Abstract. The paper demonstrates that the addition of automatically selected word-pairs substantially increases the accuracy of text classification which is contrary to most previously reported research. The word-pairs are selected automatically using a technique based on frequencies of n-grams (sequences of characters), which takes into account both the frequencies of word-pairs as well as the context in which they occur. These improvements are reported for two different classifiers, support vector machines (*SVM*) and k-nearest neighbours (*kNN*), and two different text corpora. For the first of them, a collection of articles from PC Week magazine, the addition of word-pairs increases micro-averaged breakeven accuracy by more than 6% point from a baseline accuracy (without pairs) of around 40%. For second one, the standard Reuters benchmark, SVM classifier using augmentation with pairs outperforms all previously reported results.

1 Introduction

Text classification is the problem of automatically assigning electronic text documents to pre-specified categories. In recent years, many statistical classification and machine learning techniques have been successfully applied to this problem [1,7,10,18,19]. Most of these systems use a simple "bag of words" representation of text in which each feature corresponds to a single word. Typically, this results in feature sets consisting of hundreds of thousands of features. Many learning algorithms such as decision trees, Bayes probabilistic models and neural networks have serious difficulty dealing with such large feature spaces. Hence, earlier research on feature extraction for text classification has primarily focussed on reducing dimensionality of input space [14,19].

In contrast, in this paper, we investigate the effect of increasing the size of the native input feature space by augmentation with headline features and second order features. In particular, we focus on the addition of word-pairs that are derived automatically using a technique based on frequencies of n-grams or sequences of characters, and investigating the effect of such addition on text classification performance. We note that the use of *linguistically* derived phrases have been found to be of limited utility in text classification [7,9,13,16], while *statistically* derived word sequences have been found to be useful in classification of web pages and newsgroup data [15,8]. Our investigation into word-pairs selected on the basis of n-gram frequencies confirms that the addition of word pairs does give better accuracy, and moreover, the larger the proportion of word-pair features added, the larger the gain.

The paper is organised as follows. Section 2 describes the n-gram-based feature selection technique that is used to extract word-pairs. Section 3 discusses two classifiers that are used for categorisation: k-nearest-neighbour classifier and support vector machines. In Section 4, we describe the experimental setup: the document corpora used for empirical validation and the evaluation metrics. We then present the experiments and their results in Section 5, and discuss their implications in Section 6. Section 7 summarises our contributions.

2 *n*-Gram Method for Feature Selection

The aim of the feature selector introduced in this section is to select terms which are good discriminators of the categories of interest. Here the terms are either words or word pairs, and n-grams are only the means for the selection of these terms.

An n-gram is a sequence of n consecutive characters in a document generated by sliding a "window" n characters wide across the document text, moving it one character at a time. For example, the sequence of characters "to build" will give rise to the following 5-grams: "to bu", "o bui", " buil" and "build". Thus, n-grams may span across words.

The n-grams have several advantages over words as basic units for text processing. In particular, they are language-independent and hence, the same software can be used to process documents in any language. They also permit operation with degraded text - documents that contain typos and other transcription errors [11]. Due to these advantages, n-grams have been used for text indexing [2], for text filtering [3], text categorisation [4] and for index term extraction [6].

The idea of our approach to selecting discriminating features for a category is based on [6] and its main elements are as follows.

– First, the distribution of n-grams is computed by counting the occurrence of n-grams inside and outside the category. On the basis of this count, each n-gram is assigned *a novelty score* per category that indicates how novel or unique it is for the category.
– The novelty score of each n-gram is then apportioned across its characters, in a way that the middle character gets a higher score than the characters surrounding it, e.g., for $n = 5$, the score is distributed across the 5 characters in the ratio 0.05 : 0.15 : 0.60 : 0.15 : 0.05, respectively. This apportioning allows each character in the documents in the category to be assigned a weight that is the sum of the contributions from all the n-grams that contain this character. For instance, the character "o" in the word "score" in the context "to score by", will have contributions from the 5-grams, "o scor", " scor", "score", "core ", and "ore b". The contributions depend both on the n-gram score as well as the position of that character in the n-gram.

- Next, each instance of a word or phrase is assigned a score which is the *average* of the component character weights. Thus, there is no bias towards longer or shorter words. Since the same word may occur in different contexts and thus have different contributing n-grams, different instances of the same words may have different scores. Hence, the final score of a feature for the category is computed as an *average* of the scores for different instances of the same feature in the category.
- Finally, these scores are used to rank the features and eliminate those with scores less than a *cut-off threshold* (= mean plus one standard deviation).

In our experiments we have used $n = 5$. The novelty score Ψ_i for the i-th n-gram is allocated according to the following developed in [6]:

$$\Psi_i = \begin{cases} C_i \ln \frac{C_i}{C} + B_i \ln \frac{B_i}{B} - (C_i + B_i) \ln \frac{C_i+B_i}{C+B} & \text{if } \frac{C_i}{C} \geq \frac{B_i}{B}, \\ 0 & \text{otherwise,} \end{cases}$$

where C_i and B_i are number of occurrences of the n-gram, and C and B are the total number of n-grams in the category and in the background, i.e. outside of it, respectively. The non-zero component in the above formula is the G^2 statistics, and has a large value only when $\frac{C_i}{C}$ is much greater than $\frac{B_i}{B}$, i.e., when the n-gram is much more likely to occur in the category than in the background.

3 Classifiers

In our experiments we have used *k-nearest-neighbour* classifier (kNN) and *support vector machines* (SVM). This choice was primarily due to the fact that both kNN and SVM scale well to large feature spaces (unlike, e.g., decision trees or back propagation neural networks)[1]. This is particularly important here since the feature space we are dealing with is large to start with and the addition of pairs will make it larger still. Other properties that make these two classifiers very suitable for this work are:

- Both kNN and SVM can naturally provide a score that indicates the likelihood that a document belongs to a given category. These scores may be used for ranking documents, and measuring precision at different recall levels.
- kNN and SVM differ statistically: kNN is a non-parametric[2] and non-linear classifier, while our SVM is a linear parametric model.
- kNN and SVM are both easy to implement.

For both classifiers, each document is represented as a sparse *binary* vector of feature presence, i.e. with entries indicating if a word or word-pair is present

[1] Our aim is not to find the ultimate accuracy, but to demonstrate the usefulness of addition of pairs. To that end, those two classifiers selected should provide an indicative sample.

[2] We mean non-parametric in the sense that no parameters are estimated from the training data.

or not. This representation was chosen due to its simplicity and due to the fact that, in preliminary tests with both classifiers, it performed on par with more complicated frequency counts representation (cf. similar observation in [7]).

We have used the linear kernel SVM in our experiments [5,17], with a separate machine trained for each category. The output for a category c is a ranked list of documents with scores allocated according to the following formula:

$$score_{SVM}(d) = w \cdot x + b,$$

where x is the vector of features for document d, "$\cdot$" denotes the dot product in features space and the vector w and bias b are determined by minimising the following functional:

$$\Phi(w, b) = \frac{1}{2}(||w||^2 + b^2) + C \sum_{i=1}^{n} \max\Big(0,\ 1 - y_i(w \cdot x_i + b)\Big)^2,$$

where $(x_i, y_i) \in R^m \times \{-1, 1\}$ are labelled training instances, i.e. the feature vector for document d_i and corresponding label, where 1 indicates membership of the category and -1 indicates non-membership, n is the number of training instances and $\xi \mapsto \max(0, \xi)^2$ is the error penalty function. The constant C controls the trade-off between the complexity of the solution and the error penalty. It has been set to $C = 0.850$ for all our tests (this value is not critical).

The score for kNN is based on the similarity between the test document d and the k nearest neighbours in the training document set, and whether these neighbours are in category or not. It is equal to:

$$score_{kNN}(d) = \sum_{i \in kNN(d)} y_i \frac{x \cdot x_i}{||x||\ ||x_i||},$$

where $kNN(d)$ are indices of k nearest neighbours of the document d in terms of the "cosine" similarity, $Similarity(d, d_i) \rightarrow x \cdot x_i/(||x||\ ||x_i||)$, and, as above, x_i and y_i are the ith training document feature vector and label, respectively. The value for k for our experiments was chosen to be 30 after careful experimentation.

4 Experimental Setup

In our experiments we first pre-process the data, determine the feature sets for different experimental settings, and then use the two selected classifiers to learn models of different categories. This is accomplished using the training set documents only. The created models are then evaluated on the test sets using micro-averaged breakeven point [12] that has been the benchmark measure for the Reuters data set.

4.1 Data Collections

Our first data corpus is the *Reuters*-21578 news-wires collection of documents. This is the most commonly used corpus in text classification research. In particular, the modApte split, available at http://www.research.att.com/lewis, has been used as a standard benchmark in [1,7,10,18]. This split has 9603 training documents and 3,299 test documents. We have used 115 categories, i.e. all those with at least one training case, while some researchers have used 95 categories, i.e. all those with at least two training cases. The categories cover topics such as commodities, interest rates and trade. The frequency of occurrence of categories varies greatly. For instance, roughly 30% of the documents appear in *earnings*, while *ringgit*, *rubber* and *rupiah* have only one training example each.

The second corpus used is *ComputerSelect* collection, consisting of a set of 6070 articles from *PC Week* magazine extracted from the ComputerSelect CD December 1996. These articles have manually assigned categories with many articles belonging to multiple categories. There are 840 categories, out of which the largest 128 were chosen for our tests. Only those articles that belonged to one of these 128 categories are included in the document set, which resulted in a total of 5257 articles. This set was then split into 2807 training articles and 2450 test articles such that all 128 categories contained at least 5 training articles. Again, as with the Reuters collection, the frequency of occurrence of categories varies greatly. For example *Software Product Introduction* appears in 10% of the documents while *Protocol Gateway Software* has only 20 documents assigned to it. This collection is more of a real-world example of classification since the manual assignments to categories have not been evaluated by multiple people, and not all categories are based on subject matter alone. Some categories indicate type of article, e.g., *Software Product Introduction*, while others indicate what the article is about, e.g., *Image Scanner.*

4.2 Pre-processing

First, all text is converted to a single case and non-alphabetic characters are turned into spaces, and then the sequences of consecutive blanks are compressed into a single space. Next, words that are in a standard stop list are removed, and the remaining words stemmed using the standard rule-based Porter stemmer. For instance, the text "for building, Permits" will be converted to "build permit". The text is then processed to yield different feature sets for different experiments.

4.3 Performance Measure

The standard evaluation criterion for the Reuters benchmark is the *breakeven point*, the point at which precision equals recall. Since this is different for different categories, the standard evaluation figure is the *micro-averaged breakeven point* (μBP) [12]. Micro-averaging gives each category-document assignment decision equal weight. We have some concerns over the use of this single figure of merit, particularly since the micro-averaging step can obscure many subtle performance details. However, this figure has been widely reported by a number of researchers, and hence we report the same for the sake of comparison.

5 Experiments

In this paper, we focus on the investigation of the effect of augmentation of input feature space with word-pairs. Hence, in each of the following settings, we first learn models using a baseline feature set. Then the accuracy of each model is compared with the performance of models learnt by augmenting this baseline feature set with the set of word-pairs extracted using the n-gram feature selector.

Two set of experiments have been conducted, one with universal dictionaries, i.e. with all categories using the same feature set, and the other with category specific dictionaries, "optimised", i.e. selected, for each category separately.

5.1 Experiments with Universal Feature Sets

The following four baseline feature sets are considered for augmentation with word-pairs.

1. All words: The data collection is pre-processed, and all words from the training set documents are used as baseline features. This gives 20197 words for Reuters and 20572 words for ComputerSelect.
2. Selected words: We first create category-specific word sets as follows. For each category, up to 300 words describing "in-category" documents are selected using the n-gram feature selector (cf. Section 2). Then the same number of words describing "out-of-category" documents are added. In general, for most categories, this results in far fewer than 300 "in-category" words (since many words do not have a score larger than our cut-off threshold see Section 2). There is also an overlap between "in-category" and "out-of-category" words. All this results in the largest category-specific word set having 176 items for Reuters and 359 for ComputerSelect.
   The universal word sets are created by pooling all the category-specific word sets together. For Reuters, this results in a baseline feature set of 1375 words, while the ComputerSelect collection consists of 2552 words. These consist of 6.8% and 12.4% of the total word sets, respectively.
3. All words + headlines: In this setting, the baseline feature set in the first setting is augmented with words extracted from headlines in the training set. All words other than those in a standard stop list are retained. The number of such headline words for Reuters set is 8179 while the ComputerSelect set contains 3403 headline words.
4. Selected words + headlines: In this setting, the same headline words extracted in the previous setting are used to augment the features in the "selected words" setting as above.

The universal word-pair set is selected using the n-gram feature selector in the same manner as the universal word set. For Reuters, we selected 2714 pairs from the total set of 303086, while for ComputerSelect we obtained 4050 pairs from the total of 413572 word-pairs in the training set. Thus, less than 1% of the word-pair feature space is selected by the n-gram feature selector.

Table 1. Micro-averaged breakeven point (μBP %) for the *universal feature* sets experiments illustrating the impact of addition of pairs

Experimental Setting	Number Baseline Features	μBP % for **SVM**			μBP % for **kNN**		
			Sel. Method			Sel. Method	
			n-gram	χ^2		n-gram	χ^2
Reuters		**Base.**	**+2714 Pairs**		**Base.**	**+2714 Pairs**	
All words	20197	85.6	86.3	85.8	75.6	76.3	75.8
Selected words	1375	85.6	86.2	85.7	77.1	78.3	77.5
All words + headlines	28376	86.8	87.3	87.0	77.2	77.9	77.3
Sel. words + headlines	9554	87.0	87.5	87.1	78.2	79.3	78.4
ComputerSelect		**Base.**	**+4050 Pairs**		**Base.**	**+4050 Pairs**	
All words	20572	39.6	45.6	45.8	28.4	32.9	32.5
Selected words	2552	38.6	45.9	45.8	30.5	35.0	35.1
All words + headlines	23124	40.6	45.9	46.3	29.1	33.5	33.2
Sel. words + headlines	5595	40.1	46.2	46.0	31.0	35.2	35.3

Table 1 lists the micro-averaged breakeven point for the different experiments conducted for the two classifiers for the 115 Reuters categories and 128 ComputerSelect categories. For the purpose of comparison with other feature selection techniques, results are also reported for the situation where the same number of word-pairs are selected using maximum of χ^2 statistics, which was the best word selector in [19].

As expected, the breakeven points for the real world collection of ComputerSelect articles are much lower than that for Reuters. As evident from Table 1 both classifiers show improvement in their performance once selected word pairs are added (cf. Section 6 for further discussion). The accuracy for kNN is lower than that for SVM for both collections

5.2 Category Specific Feature Sets

Another approach to feature selection for text classification is the use of small category-specific feature sets, which has been found to increase accuracy [7]. In this context, it is useful to investigate the effect of the addition of category-specific word pairs to category-specific word sets. The feature sets for this investigation are created as described in Section 5.1. The average size of the word set for Reuters and ComputerSelect is 55.4 words and 107.2 respectively. However, there is a large variation in the size of the word sets, e.g. category "*earnings*" in Reuters with the largest number of training examples has a feature set with 176 words, while some categories (with one training example) have an empty word set. For ComputerSelect, the feature set size varies between 25 to 359. Classification performances are compared by adding category-specific pair sets which are generated as described in Section 5.1. These sets also show enormous variation for different categories. For Reuters the average, maximum and minimum sizes of the pair set are 55.4, 352 and 0 respectively, while for ComputerSelect the average is 88.2 pairs and the sizes vary from 6 to 257.

Table 2. Micro-averaged breakeven point (μBP %) for the *category-specific* feature sets experiments illustrating the impact of addition of pairs

Base Data	No. of Base Feat.	μBP % for **SVM**		μBP % for **kNN**	
Reuters	**Min–Max**	**Baseline**	**+ 0–352 Pairs**	**Baseline**	**+ 0–352 Pairs**
Word sub-sets	0–176	84.3	85.4	78.6	79.4
Word sub-sets + headlines	8179–8355	87.9	88.8	80.3	80.4
ComputerSelect	**Min–Max**	**Baseline**	**+ 6–257 Pairs**	**Baseline**	**+ 6–257 Pairs**
Word sub-sets	6–257	39.6	46.8	33.1	38.3
Word sub-sets + headlines	3409–3660	41.2	48.1	29.9	35.5

Table 2 shows the micro-averaged breakeven point for experiments with category specific word sets. As before the addition of pairs does indeed improve the categorisation performance in all cases. Comparing these results with the results for the selected word set in Table 1, we see that the use of category specific word and pair sets instead of the larger pooled sets does not substantially decrease performance, and in many cases it actually improves it.

6 Discussion

As shown in Tables 1 and 2, the statistically-derived word pairs, in general, increase the categorisation performance for both classifiers and corpora. Below we analyse the significance of observed improvements and different contributing factors.

Differences in baseline precisions: Baseline precisions obtained for ComputerSelect are much lower than for Reuters (e.g. $\tilde{4}0$% vs. $\tilde{8}5$% for SVM classifier, respectively). This can be attributed to qualitative differences in the documents in the two collections: Reuters articles are written by journalists to disseminate information and hence contain precise words that are useful for classification, e.g., "grain" and "acquisition", whereas ComputerSelect articles are written as advertisements and contain words, such as, "software" and "system", which are useful only in context, e.g., "network software" and "array system".

Gain relative to ideal classifier: In the universal set situation, for ComputerSelect we observe over 6% point improvement from a low base (40% for SVM), while the increase in precision for Reuters is less than 1% point from a high starting base (85% for SVM). A fairer comparison which takes into account the different baselines can be made by calculating relative gain, $\frac{\mu BP(new)-\mu BP(old)}{100-\mu BP(old)}$, which compares the observed improvement to the ultimate improvement possible for the ideal classifier. When measured this way, the improvement for both Reuters (4% on average) and ComputerSelect (10% for SVM and 6% for kNN) is of similar, substantial size.

Significance test: We have performed a matched pairs t-test, where for each setting of document collection, classifier and baseline feature set, we paired the breakeven point for each category using just the baseline set with the corresponding breakeven point after the addition of pairs. In each case our conjecture is that the use of pairs provides an improvement in performance, hence the alternative hypothesis (H_1) is that the mean breakeven point for feature sets with pair features is greater than that for the corresponding set without pair features, i.e., $\mu_p > \mu_b$, where μ_b and μ_p are the mean breakeven points for the baseline feature set and for the baseline set plus pairs, respectively. The null hypothesis (H_0) is that there is no difference in the mean breakeven point between the baseline feature set and the set with pair features, i.e. $\mu_p = \mu_b$. Using a one tailed t-test, in all except one case (Reuters, SVM, selected set + headlines) we can reject the null hypothesis with a significance level of 1%.

Note that the t-test performed gives equal importance to all categories, independent of their sizes. In contrast, the improvements measured relative to the ideal classifier based on μBP gives each category the importance proportional to its size. This also shows substantial improvements, as discussed above.

Larger proportion of pairs, larger accuracy improvements: In order to analyse which categories are helped by the addition of pairs, Figure 1 plots the difference between micro-averaged breakeven points with pairs and without pairs for the universal dictionaries as the number of categories is increased. The categories are ordered in the descending order of their training set sizes.

Referring to Figures 1(b) and 1(c), which show the strongest difference in performance for different baseline sets, the order of the curves suggests that the increase in performance due to pairs increases as the size of baseline set decreases. For instance, the lowest curve in Figures 1(b) and 1(c) belongs to the largest baseline feature set, namely, all words + headlines, while the highest curve belongs to smallest baseline set, namely, selected word set. This observation may explain why earlier experiments using a small number of linguistically derived phrases produced little improvement [7,9,13,16].

In fact, a closer analysis of Reuters results shows that n-gram selected pairs provide non-positive gains in breakeven point if the ratio of the number of the (category-specific) selected word-pairs to selected words is $< 70\%$ and non-negative gains if otherwise (with only 3 exceptions!).

Significant improvement for medium size categories: The curves in Figure 1(a) and (b) indicate that pairs provide greatest improvement in classification accuracy for medium sized categories for Reuters. The larger ComputerSelect categories are similar in size to the medium sized categories for Reuters given that the largest category for the ComputerSelect document set is only 10% of the data and consists of 287 examples while the largest category in Reuters contains nearly 30% of the data ($\sim$3000 training examples). Not surprisingly, the performance on the ComputerSelect collection improves for all categories when pairs are taken into account (cf. Figure 1(c) and (d)), unlike the Reuters case where the difference is small and sometimes negative for the larger categories.

In fact, for the medium size Reuters categories (numbered 12 to 45 in the decreasing order) the ratio of pairs to selected words is $> 100\%$ (with 2 exception),

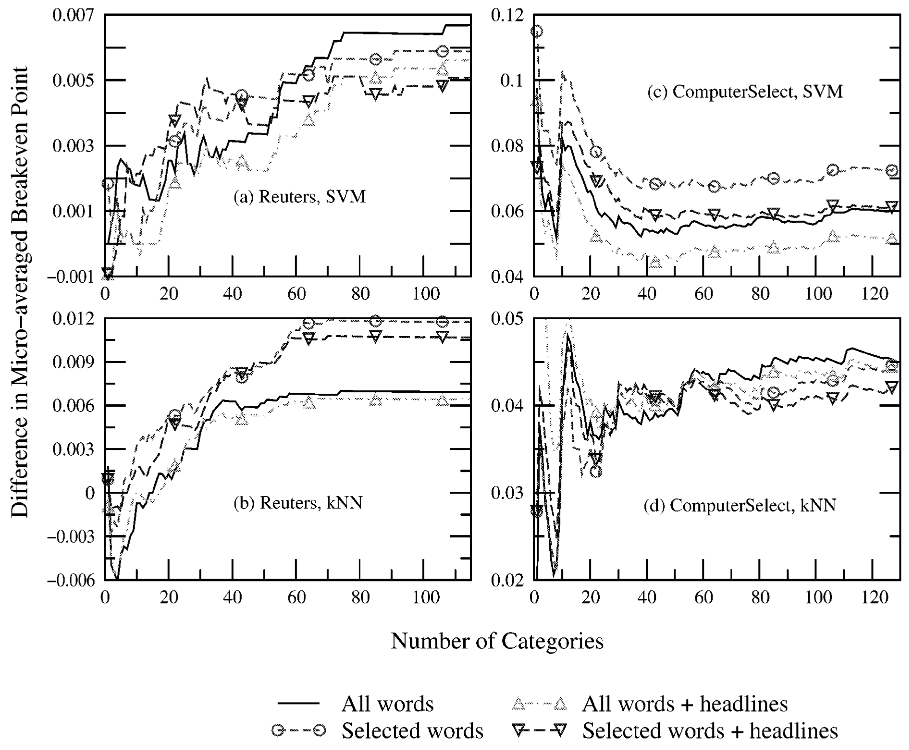

Fig. 1. Influence of Pairs on Classification Performance. The figures show the difference between micro-averaged breakeven accuracy with pairs and without pairs as a function of the number of categories included. Here the categories are ordered in the descending order of their training set sizes.

and the gain in breakeven point is ≥ 0, with very solid positive improvements noted for the (majority of) categories numbered 14 to 28.

Comparisons of n-gram and χ^2 word-pair selectors: Addition of pairs selected by either method shows improvements (cf. Table 1). For Reuters, using SVM, just 2714 selected pairs with 7% of words produces better μBP than all 20197 words. For ComputerSelect using SVM, just 4050 pairs more than compensates for the 87.5% of the words that were discarded. However, the improvement for pairs selected by χ^2 statistics is smaller in most cases than for n-gram method.

The n-gram method has other important advantages. For instance, this method produces word sequences of any length m with just two passes over the input, while other methods based on direct, explicit ranking of features, need m passes in such a case [15,8]. Further, for large document corpora, the number of n-grams is smaller than the number of word-pairs (in the two collections considered, the number of word-pairs is more than double the number of n-grams), and hence, n-gram method is also more memory efficient. Finally, although we used language-dependent techniques such as stemming, the n-gram method itself is designed to be language-independent since the basic unit for measuring novelty is just sequences of characters.

Use of category-specific feature sets: The use of small category-specific feature sets has been investigated by Dumais et. al [7]. They use 300 category-specific features with SVM classifier and obtained high classification accuracy. Our experiments show that category-specific word sets combined with category-specific pair sets can lead to higher accuracy than using universal sets with or without pairs. The result on the Reuters set for the SVM classifier using category specific word and pair sets on all 115 categories outperforms all previously reported results on this benchmark. The best earlier performance on the same data set is 87.8% for the top 95 categories and is obtained using a very complex classifier consisting of an ensemble of decision trees [18] and uses words and headline features. This figure is less than our figure of 88.78% for all 115 categories. Our best performance for the top 95 categories is 88.9%. This translates to the gain of 9% relative to the hypothetical ideal classifier.

7 Conclusion

We have presented a method (n-grams) for automatically selecting phrases that can take into account both the co-occurrent frequencies as well as the context in which the pairs occur. We have shown that addition of such pairs to the input feature space substantially improves performance of classifiers (kNN and SVM in our experiments). This is valid for at least two different collections examined in this research with very different properties and very differing baseline precisions (40% for ComputerSelect and 85% for Reuters). The improvements translate to 4-10% gain of relative to the hypothetical ideal classifier. In particular, for the standard Reuters benchmark we obtained SVM classifiers (using augmentation with selected pairs) outperforming all previously reported results.

Our results show significant improvements occurring mainly for medium size categories. We found that in general the larger the proportion of pairs added to word sets the larger the improvement in performance suggesting the usage of small baseline sets augmented with pairs for compact accurate classifiers.

Our future work will focus on investigation of the effect of the relative sizes of the baseline and augmenting set on the classification accuracy, and evaluation of the usefulness of pairs for automatic email and news categorisation.

Acknowledgements. The permission of the Chief Technology Officer, Technology Strategy and Research, Telstra Corporation, to publish this paper is gratefully acknowledged.

References

1. C. Apte, F. Damerau, and S. M. Weiss. Text Mining with Decision Trees and Decision Rules . In *Conference on Automated Learning and Discovery*, 1998.

2. J. E. Burnett, D. Cooper, M. F. Lynch, P. Willett, and M. Wycherley. Document Retrieval Experiments Using Indexing Vocabularies of Varying Size. I. Variety Generation Symbols Assigned to the Fronts of Index Terms . *Journal of Documentation*, **35**(3):197–206, (1979).
3. W. B. Cavnar. N-gram-based text filtering for TREC-2 . In *Proceedings for Second Text Retrieval Conference (TREC-2)*, pages 200–215. NIST Special Publication, 1993.
4. W. B. Cavnar and J. M. Trenkle. N-gram-based text Categorization . In *Proceedings of Third Annual Symposium on Document Analysis and Information Retrieval*, 1994.
5. N. Christianini and J. Shawe-Taylor. *Support Vector Machines and other Kernel Based Methods* . Cambridge University Press, 2000.
6. J. D. Cohen. Highlights: Language- and domain-independent automatic indexing terms for abstracting . *Journal of the American Society for Information Science*, **46**(3):162–174, (1995).
7. S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive Learning Algorithms and Representations for Text Categorization . In *Seventh International Conference on Information and Knowledge Management*, 1998.
8. J. Furnkranz. A Study Using n-gram Features for Text Categorization . Technical report, Austrian Reserach Institute for Artificial Intelligence, 1998.
9. J. Furnkranz, T. Mitchell, and E. Riloff. A Case Study in Using Linguistic Phrases for Text Categorization on the WWW . In *In AAAI-98 Workshop on Learning for Text Categorization*, 1998.
10. T. Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features . In *Proceedings of the Tenth European Conference on Machine Learning ECML98*, 1998.
11. K. Kukich. Technique for automatically correcting words in text . *ACM Computing Surveys*, **24**:377–439, (1992).
12. D. Lewis. Evaluating Text Categorization . In *Proceedings of the Speech and Natural Language Workshop*, 1991.
13. D. Lewis. An Evaluation of Phrasal and Clustered Representations on a Text Categorization Task . In *Proceedings of the Fifteenth International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1992.
14. D. Lewis. Feature selection and feature extraction for text categorization . In *Proceedings of the Speech and Natural Language Workshop.* Defense Advanced Research Projects Agency, 1992.
15. D. Mladenic and M. Grobelnik. Word Sequences as Features in Text Learning . In *In Seventeenth Electrotechnical and Computer Science Conference*, 1998.
16. S. Scott and S. Matwin. Feature Engineering for Text Classification . In *Proceedings of the Sixteenth International Conference on Machine Learning*, 1999.
17. V. Vapnik. *Statistical Learning theory* . Wiley, 1998.
18. S. M. Weiss, C. Apte, F. Damerau, D.E. Johnson, F. J. Oles, T. Goetz, and T. Hampp. Maximizing Text-Mining Performance . *IEEE Intelligent Systems*, **14**(4), (1999).
19. Y. Yang and J. O. Pedersen. A Comparative Study on Feature Selection in Text Categorization . In *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997.