

Automatic Textual Document Categorization Using Multiple Similarity-Based Models

Kwok-Yin Lai^{*} and *Wai Lam*[†]

Abstract We develop a similarity-based textual document categorization method called the generalized instance set (GIS) algorithm. GIS integrates the advantages of linear classifiers and k-nearest neighbour algorithm by generalization of selected instances. To further enhance the performance, we propose a meta-model framework which combines the strength of different variants of GIS algorithm as well as state-of-the-art existing algorithms using multivariate regression analysis on document feature characteristics. Document feature characteristics, derived from the training document set, capture some inherent properties of a particular category. Different from existing categorization methods, our proposed meta-model can automatically recommend a suitable algorithm for each category based on the category-specific statistical characteristics. In addition, our meta-model differs from existing multi-strategy learning in that our approach is not limited to the number and type of component classifiers. By flexible addition and substitution of different classifiers, incremental classification performance can be obtained. Extensive experiments have been conducted. The results confirm that our meta-model approach can exploit the advantage of its component algorithms, and demonstrate a better performance than existing algorithms.

^{*}Corresponding Author: Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Shatin, Hong Kong. {kylai@se.cuhk.edu.hk}

[†]Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Shatin, Hong Kong. {wlam@se.cuhk.edu.hk}

1 Introduction

The aim of document categorization is to assign a number of appropriate categories to a text document. The goal of text categorization is to learn a classification scheme that can be used to classify text documents automatically.

There have been researches conducted for this automatic text categorization task. Apte et al. [1] adopted a decision tree learning technique to learn a classifier in the form of a decision tree. Yang and Chute [16] proposed a statistical approach known as Linear Least Squares Fit (LLSF) which estimates the likelihood of the associations between document terms and categories via a linear parametric model. Lewis [10] explored linear classifiers for the text categorization problem. Cohen and Singer [3] developed the sleeping experts algorithm which is based on a multiplicative weight update technique. Yang [14] developed an algorithm known as ExpNet which derives from the k-nearest neighbor technique. ExpNet achieves good categorization performance on large document corpora such as the Reuters collection and the OHSUMED collection [18]. Lam, Low and Ho [8] attempted to tackle this problem using Bayesian networks. Joachims [6] and Yang [17] recently compared support vector machines with k-NN, while Dumais et al. [4] compared support vector machines, decision trees and Bayesian approaches on the Reuters collection.

We propose a new technique known as the generalized instance set (GIS) algorithm by unifying the strength of k-NN and linear classifiers. We have implemented our GIS algorithm, the ExpNet algorithm, and some linear classifiers. SVM [12] is also included in our experiments for comparative study, as it demonstrated to have good performance in recent studies [6, 17]. Extensive experiments have been conducted on two document corpora, namely the OHSUMED collection which contains medical journal abstracts and the Reuters-21578 collection. The results show that our new approach outperforms k-NN, linear classifiers and SVM in most of the experiments.

To further enhance the performance, we propose a meta-model framework which combines the strength of GIS algorithm as well as state-of-the-art existing algorithms using multivariate regression analysis on document feature characteristics. Document feature characteristics, derived from the training document set, capture some inherent properties of a particular category. Different from existing categorization methods, our proposed meta-model can automatically recommend a suitable algorithm for each category based on the category-specific statistical characteristics. The results confirm that our meta-model approach can exploit the advantage of its component algorithms, and demonstrate better performances than existing algorithms such as k-NN, SVM and linear classifiers, namely WH and Rocchio.

2 Similarity-Based Categorization

A document basically consists of texts is usually represented by a vector of features which are derived from the words appeared in the document. This feature-based representation will be used for training and on-line categorization. Documents can be assigned to more than one category, and each training document has the

known categories assigned to it. A distinct classification scheme is automatically constructed from training documents by machine learning for each category. Each classification scheme is able to determine the membership of a document to a particular category. Each element in the vector denotes the weight associated with a particular feature. Features typically are terms composed of individual words or phrases automatically extracted from the documents collection. Once the set of schemes for all categories have been constructed, they can be used simultaneously to determine the set of appropriate categories for each document¹.

We denote D_j as an instance in the training collection and it is represented as (a_{1j}, \dots, a_{nj}) , and denote X as a request document to be categorized and it is represented as (x_1, \dots, x_n) .

Some recent document classification learning approaches can be regarded as similarity-based algorithms. In these algorithms, each document is mapped to an internal representation. A metric measuring the similarity of two documents is then designed. This similarity metric is used during the training phase as well as in the online classification. Two common similarity-based document classification algorithms have been discussed, namely the k-NN algorithm [14] and linear classifiers [7, 13].

3 The Generalized Instance Set Algorithm (GIS)

We propose a new technique called the generalized instance set (GIS) algorithm which unifies the strength of the k-NN algorithm and linear classifiers and taking into account some characteristics of document classification. The main idea is to construct a set of *generalized instances* (GI) to replace the original training examples. Given a particular category, it can be observed that the regularity among positive examples is usually more explicit than that of negative examples. The pattern or classification knowledge induced from a pool of similar positive examples are relatively accurate. However, negative examples close to such pool are likely noise. By selectively substituting appropriate positive and negative examples in the positive example pool, we can essentially remove some noisy examples. Based on this idea, we propose the GIS algorithm which focuses on refining the original instances and constructs a set of generalized instances. The outline of the GIS algorithm is given in Figure 1.

It automatically selects a representative positive instance and performs a generalization, via the function *Generalize* in Step 9, using k nearest neighbors which may include positive and negative instances. A generalized instance G is formed after the generalization process. This G will be evaluated by the function *Rep* denoting the representative power. If the representative power of G is better than the old one (i.e. G'), we use G as a new point and repeat the search and generalization task again. The algorithm will continue to search for the best local generalized instance as illustrated from Step 7 to Step 12 in the algorithm. If there is no further

¹For k-NN algorithms, a single classification scheme for all categories can be implemented to conduct categorization. However, for analysis purpose it is equivalent to having a separate classifier for each category as explained later in the paper.

Input: The training set T
The category C

Procedure GIS(T, C)

- 1) Let G and G' be generalized instances.
- 2) GS be generalized instance set, and GS is initialized to empty.
- 3) Repeat
- 4) Select a positive instance as G .
- 5) Rank instances in T according to the similarity metric with G .
- 6) Compute $Rep(G)$.
- 7) Repeat
- 8) $G' := G$.
- 9) $G := Generalize(G', k)$.
- 10) Rank instances in T according to the similarity metric with G .
- 11) Compute $Rep(G)$.
- 12) Until $Rep(G) < Rep(G')$
- 13) Add G' to GS .
- 14) Remove top k instances from T .
- 15) Until no positive instances in T .
- 16) Return GS .

Figure 1. *The Generalized Instance Set (GIS) algorithm*

improvement in terms of the representative power, the last generalized instance G' is added to the generalized instance set GS and the corresponding k nearest neighbors are removed from the training document collection. This process is repeated until no positive instance remains in the training document collection. As the learning progresses, it constructs a number of generalized instances and stores them in GS .

The representative power function $Rep(G)$ for a generalized instance G is defined as follows:

$$Rep(G) = \sum_{I^+ \in K} (k - rank(I^+))$$

where K is the set of k nearest neighbors of G , I^+ is a positive instance in K and $rank(I^+)$ denotes the ranking of the instance I^+ in the set K according to the similarity metric. Large value for $Rep(G)$ implies that more positive instances are found in the set of k nearest neighbors of G .

A variety of methods can be used for the generalization task in Step 9. We have tried two methods based on the Rocchio and Widrow-Hoff (WH) algorithms. Let P_k and N_k be the set of positive and negative instances in k nearest neighbors of G respectively. The generalization process based on the Rocchio algorithm is given as follows:

$$G' = \frac{\sum_{I \in P_k} I}{|P_k|} - \eta \frac{\sum_{I \in N_k} I}{|N_k|}$$

where η is the parameter that adjusts the relative impact of positive and negative neighboring instances. The summation is taken as the vector addition. $|S|$ denotes the cardinality of the set S .

The generalization function based on the WH algorithm processes each instance in k nearest neighbors of G one by one in each iteration. Let G_i denote the intermediate value of the generalized instance at the i -th iteration. Initially, the elements in the generalized instance is set to all zeros denoted by $G_0 = \vec{0}$. At each iteration, G_{i+1} is computed from G_i and the current instance I_i .

$$G_{i+1} = G_i - 2\eta(G_i \cdot I_i - L_i)I_i$$

$$G' = G_{k+1}$$

where $\eta > 0$ is a parameter that controls how quickly G_i is allowed to change. L_i is the class label of the instance I_i . L_i is 1 if I_i is a positive instance and 0 if I_i is a negative instance. The final generalized instance G_{k+1} is the required result for G' .

The GIS algorithm learns from the training document collection and produces a classifier represented by a set of generalized instances. The classifier can then be used for classification by computing a score of a request document X . This score is computed as the weighted sum of the similarity metric of each generalized instance G . We define $Assoc(G, C)$ as the association factor between the generalized instance G and the category C . This association factor can be easily calculated during the learning phase as follows:

$$Assoc(G, C) = \frac{|P_k|}{P}$$

where P is the number of positive instances in the training set. As a result, the final score denoted by the function $Score$ for the request document X is calculated by:

$$Score(X, C) = \sum_{G \in GS} \Delta(G, X) Assoc(G, C)$$

If this score is greater than a threshold value θ , the category C is assigned to document X .

4 Experimental Results of GIS approach on Categorization

4.1 Experimental Setup

We have implemented our GIS algorithm, the ExpNet (k-NN) algorithm, the basic Rocchio algorithm and the basic Widrow-Hoff (WH) algorithm. Extensive experiments have been conducted on two large-scale document corpora, namely the OHSUMED collection and the Reuters-21578 collection.

We divided the 21,578 documents in the Reuters-21578 collection according to the “ModApte” split into a training document collection and a testing document collection. There are 9,603 training documents and 3,299 testing documents similar to the one used in Dumais et al. [4]. For each category, we used the training document collection to learn a classification scheme which is used for classifying documents in the testing document collection and compared the result with the

manual classification. We chose those 90 categories that appear in at least one training document and one testing document.

The OHSUMED collection is a bibliographical medical document collection. We used 50,216 documents in 1991 which have abstracts. There are total 14,626 distinct main headings in the OHSUMED records. In our experiment, we chose the set of 119 MeSH categories from the heart disease categories. The OHSUMED corpus is difficult to learn for a good classifier since the documents are rather noisy.

To measure the performance, we use the micro-averaged recall and precision break-even point measure (MBE) [10]. In micro-averaged recall and precision break-even point measure, the total number of false positive, false negative, true positive, and true negative are computed across all categories. These totals are used to compute the micro-recall and micro-precision. Then we use the interpolation to find the break-even point.

4.2 Results

For Reuters collection, the best performance achieved under the MBE when $\eta = 1$ for Rocchio, and when $k = 50$ for k-NN. Figure 2 depicts the MBE measures of each algorithm on all 90 categories in the Reuters-21578 corpus. It shows that GIS-W performs much better than the basic linear classifiers including Rocchio and WH. Table 1 summarizes the performance of all categories in the Reuters corpus. Using the MBE measure, GIS-W algorithm has 8.89% improvement over Rocchio, 3.05% over WH, 5.36% over k-NN and 0.48% over SVM.

For OHSUMED collection, the best performance achieved under the MBE when $\eta = 1$ for Rocchio, and when $\eta = 100$ for k-NN. Figure 3 depicts the MBE point of each algorithm on all 90 categories in the OHSUMED corpus. It shows that, while GIS-R maintains a higher recall and precision than both k-NN and Rocchio, GIS-W shows a higher recall and precision than other individual classifiers on average. Table 1 summarizes the performance of all categories in OHSUMED corpus. The last four columns show the improvement of GIS over the other four algorithms respectively. Using the MBE measure, GIS-W algorithm has 15.67% improvement over Rocchio, 5.62% improvement over WH, 9.18% improvement over k-NN and 8.16% improvement over SVM.

Corpus	Rocchio	WH	k-NN	SVM	GIS-R	GIS-W	Improvement(%)			
Reuters	0.776	0.820	0.802	0.841	0.830	0.845	8.89	3.05	5.36	0.48
OHSUMED	0.504	0.552	0.534	0.539	0.575	0.583	15.67	5.62	9.18	8.16

Table 1. Performance, measured by MBE, of all categories in the Reuters-21578 and OHSUMED corpora. The last four columns show the percentage of improvement of GIS over Rocchio, WH, k-NN and SVM respectively.

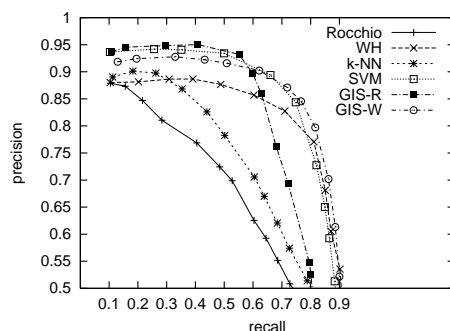


Figure 2. Micro-recall/micro-precision performance of 90 categories in the Reuters-21578 corpus.

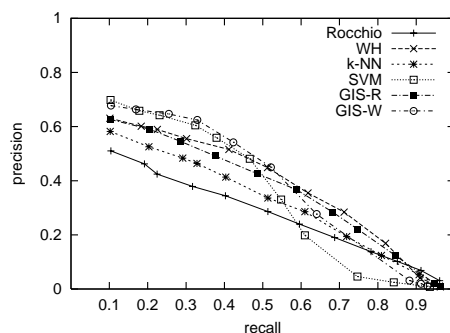


Figure 3. Micro-recall/micro-precision performance of 84 categories in the OHSUMED corpus.

5 Meta-Model Framework

5.1 Overview

The extensive research in text categorization results in a lot of different types of classification algorithms. State-of-the-art classification algorithms, such as k-NN, Rocchio, WH and SVM, have been reported to have good performances over other existing classifiers in the literature. In order to combine the strength of different classification approaches to further increase the overall performance, several recent researches have proposed multistrategy learning or combination of classifiers.

Yang et al. [15] proposed the Best Overall Results Generator (BORG) system which combined classification methods linearly, using simple equal weight for each classifier in the Topic Detection and Tracking (TDT) domain. Classification methods included Rocchio, kNN and Language Modeling. Larkey et al. [9] reported improved performance, by using new query formulation and weighting methods, in the context of text categorization by combining three classifiers, namely k-NN, relevance feedback and Bayesian independence classifiers. Instead of applying method

combination on text categorization, Hull et al. [5] examined various combination strategies in the context of document filtering. Learning algorithms included Rocchio, Nearest Neighbor, Linear Discriminant Analysis and Neural Net. Chan and Stolfo [2] presented their evaluation of simple voting and meta-learning on partitioned data, through inductive learning. Ting and Witten [11] demonstrated the effectiveness of stacked generalization for combining different types of learning algorithms. By combining a high-level model with low-level models, a better predictive accuracy was found.

Our preliminary results show that, though a particular algorithm obtains a better overall performance, it is not guaranteed that its performance is best for each category. This can be attributed to the fact that algorithms perform differently for different categories, which exhibit specific nature or different characteristics among categories. Motivated by such observation, we propose a meta-model framework for text categorization, based on multivariate regression analysis, by capturing category specific feature characteristics. We introduce document feature characteristics, which can capture some inherent properties of a particular category. Different from existing categorization methods, instead of applying a single algorithm for all categories, our new meta-model approach can automatically recommend a suitable algorithm, from an algorithm pool, for each category based on the category specific statistical characteristics.

To combine and unify the strength of different classifiers, the technique of multivariate regression analysis is integrated into our meta-model, which can automatically recommend a suitable algorithm for each category during the process of categorization. To achieve this task, we employ a learning approach by learning the relationship between the feature characteristics and the classification errors with the use of multivariate regression analysis for each algorithm. The learned relationship is expressed by sets of parameter estimates, based on which, suitable classification algorithms are recommended and applied on the validation data. Document feature characteristics, on category basis, are simply statistics which can be regarded as the descriptive summary for each category. Normalized document feature statistics are fitted into our meta-model as independent variables. The problem of predicting the expected classification error of an algorithm for a category, therefore, can be interpreted as a function of feature characteristics.

5.2 The Meta-Model Algorithm

In our meta-model, we make use of categorical feature characteristics and classification errors. In particular, we wish to predict the classification error for a category based on the feature characteristics. This is achieved by a learning approach based on regression model, in which, the document feature characteristics are the independent variables, while the classification error of an algorithm is the dependent variable. Feature characteristics are derived from the categories. Two sets of feature characteristics are collected separately from training set and tuning set. Statistics from training set are for parameter estimations. Together with the estimated parameters, the statistics from tuning set are used for predicting the classification error of an algorithm for a category. The algorithm with the minimum estimated

classification error for a category will be recommended for that category during the testing, or validation, phase. Classification errors need to undergo a logistic transformation to yield the response variable, or the dependent variable, for the meta-model. The transformation ensures that the fitted error to be in the range from 0 to 1. Consider the i th category and the j th algorithm. The response variable, y_{ij} is related to the feature characteristics by the following regression model:

$$y_{ij} = \ln \frac{e_{ij}}{1 - e_{ij}} = \beta_j^0 + \sum_{k=1}^p \beta_j^k * F_i^k + \epsilon_{ij},$$

where e_{ij} is the classification error, obtained for the i th category by using the j th algorithm. F_i^k is the k th feature characteristic in the i th category. The number of feature characteristics used in the meta-model is p . β_j^k is the parameter estimate for the k th feature, by using algorithm j . ϵ_{ij} is assumed to follow an $N(0, var(\epsilon_{ij}))$. Based on the regression model above, the outline of meta-model for text categorization is given in Figure 4.

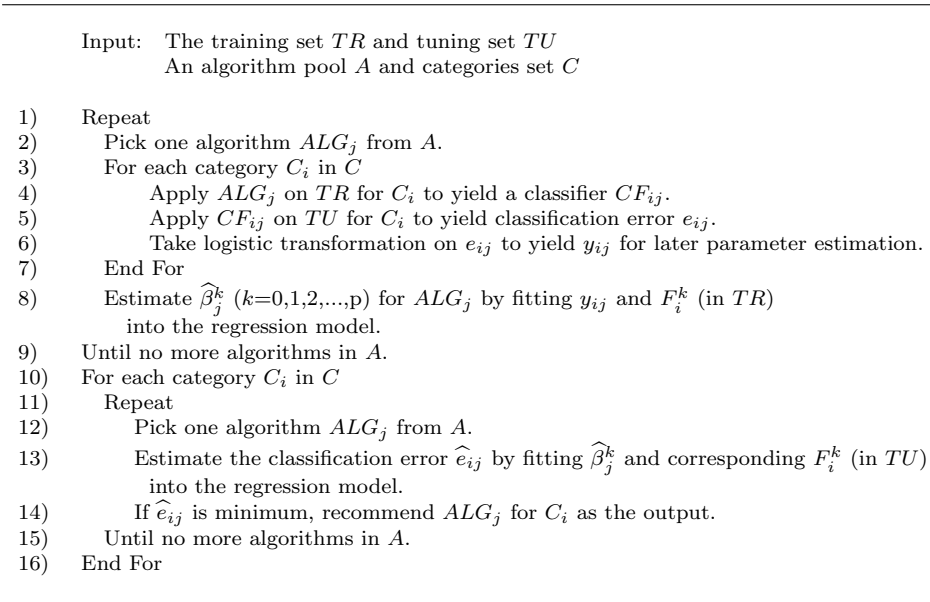


Figure 4. *The Meta-Model algorithm*

Step 1 to 9, in Figure 4, aims to estimate a set of betas ($\hat{\beta}_j^k$), the parameter estimates of the feature characteristics in the regression model, for each individual algorithm. In Step 2, an algorithm, with optimized parameter settings, is picked from the algorithm pool. By repeating Step 3 to 7, the algorithm is applied on training and tuning examples to yield classification errors of the classifier for all categories. Documents in tuning set, as shown in Step 5, are used for obtaining the classification performance, and so the classification error, of a trained classifier for each category. A set of betas, belonging to the algorithm being considered, can be

obtained by fitting all classification errors of the categories, and their corresponding feature characteristics in the training set, into the regression model. After Step 9, there will be j sets of estimated parameters, the betas, which are then used for the subsequent steps.

The predictions on the classification errors of the involved algorithms are made from Step 10 to Step 16. In Step 12, one algorithm with the same optimized parameter settings as in Step 2, is picked from the algorithm pool. The corresponding set of betas of the selected algorithm, together with the feature characteristics of a category in the tuning set, will be fitted into the regression model, in Step 13, to give the estimated classification errors of the algorithm on the category. Decisions, about which algorithm will be applied on the category, are based on the predicted minimum classification errors in Step 14. After Step 16, classification algorithms are recommended for categories, and the recommended algorithm will be applied to each category during the validation step. The robustness of the meta-model approach rests on its fully automatic estimations. The whole process, from parameter estimation to recommending algorithms for categories, is fully automatic.

6 Experiments and Results of Meta-Model

6.1 Experimental Setup

Currently, 7 feature characteristics are used in our regression model as independent variables:

1. *PosTr*: The number of positive training examples of a category.
2. *PosTu*: The number of positive tuning examples of a category.
3. *AvgDocLen*: The average document length of a category. Document length refers to the number of indexed terms within a document. The average is taken across all the positive examples of a category.
4. *AvgTermVal*: The average term weight of documents across a category. Average term weight is taken for individual documents first. Then, the average is taken across all the positive examples of a category.
5. *AvgMaxTermVal*: The average maximum term weight of documents across a category. Maximum term weight of individual documents are summed, and the average is taken across all the positive examples of a category.
6. *AvgMinTermVal*: The average minimum term weight of documents across a category. Minimum term weight of individual documents are summed, and the average is taken across all the positive examples of a category.
7. *AvgTermThre*: The average number of terms above a term weight threshold. The term weight threshold is optimized and set globally. Based on the preset threshold, the number of terms with term weight above the threshold within a category are summed. The average is then taken across all the positive examples of the category.

Two sets of normalized feature characteristics are collected separately from training set and tuning set. As illustrated in Step 8 and Step 13 in Figure 4, the feature characteristics from these two data sets serve different purposes in the meta-model: feature characteristics from training set are combined for parameters estimation, while feature characteristics from tuning set are used for predicting classification errors, base on which algorithms are recommended.

6 classification algorithms have been integrated into our meta-model. They are Rocchio, WH, k-NN, SVM, GIS-R and GIS-W, with same optimized settings as mentioned in Section 4.2. These are six recent algorithms, each of which exhibits certain distinctive nature: Rocchio and WH are linear classifiers, k-NN is instance-based learning algorithm, SVM is based on Structural Risk Minimization Principle [12] and both GIS-R and GIS-W are based on generalized instance approach.

To investigate the incremental effect after combining more and better algorithms, we have also conducted experiments by changing the size of the algorithm pool for the meta-model. Results show that there is an incremental improvement by increasing the size of algorithm pool, with different and more robust classifiers.

By using our meta-model, extensive experiments have been conducted on the Reuters-21578 corpus. Similar to the setup in Section 4, "ModApte" split is applied to divide the corpus into training set, tuning set and testing set. There are 6000 training documents, 3603 tuning documents, and 3299 testing documents. Performance is measured by micro-averaged recall and precision break-even point measure and macro-averaged recall and precision break-even point measure.

6.2 Results and Discussion

Table 1 shows that GIS-W outperforms other classification algorithms based on micro-averaged recall and precision break-even point and macro-averaged recall and precision break-even point of the ten most frequent categories respectively. Our results show that though GIS-W shows a better overall performance, the performance of other algorithms in some categories is comparable to, or even better than, that of GIS-W. This demonstrates that an algorithm, with best overall measures, does not guarantee that its performance is best for all categories. Our meta-model for text categorization can further enhance the classification performance in this aspect.

Table 2 shows the parameter estimates for the document feature characteristics of different algorithms in the meta-model. By fitting the parameter estimates and corresponding feature characteristics, on category basis, into the meta-model, the estimated classification errors of different algorithms on categories can be obtained. It should be noted that, a negative parameter estimate will lead to a smaller estimated classification error for an algorithm on a category. Besides, a feature characteristic with a large negative parameter estimate, will make itself a more important feature in voting for the algorithm than other features. For example, as seen from Table 2, *AvgTermThre* has favourable impacts for WH, GISR and GISW.

Features	RO	WH	KNN	SVM	GISR	GISW
<i>PosTr</i>	-4.75	9.24	-1.21	-0.46	5.98	9.84
<i>PosTu</i>	-0.82	-17.05	-5.88	-8.68	-17.26	-20.87
<i>AvgDocLen</i>	2028.77	3199.54	1514.31	2275.34	2567.84	2676.94
<i>AvgTermVal</i>	103.51	154.21	81.74	117.62	151.54	155.67
<i>AvgMaxTermVal</i>	14.04	23.21	12.37	15.59	16.95	21.39
<i>AvgMinTermVal</i>	-69.79	-104.37	-68.77	-92.12	-124.89	-138.73
<i>AvgTermThre</i>	-2006.50	-3164.39	-1495.49	-2250.07	-2534.19	-2640.33

Table 2. Parameter estimates for document feature characteristics of different algorithms.

Besides of comparing the performance of our meta-model over individual component algorithms, we are also interested in investigating how good we are doing when compared with the ideal combination of algorithms, which we set up as another benchmark of the meta-model performance. The ideal combination of algorithms is composed of the best algorithms, which outperform all the other algorithms within a category. Table 3 depicts the selected algorithms and their performance within a category by meta-model (M-Model) and the ideal combination (IDEAL), for the ten most frequent categories. Meta-model can estimate the ideal algorithms (in bold) correctly for 6 categories out of the 10 most frequent categories. For the remaining 4 categories, our meta-model can estimate the second best algorithms. Our results, not shown in the table, show that the meta-model can identify the ideal algorithms for 60 categories out of the total 90 categories.

Category	RO	WH	KNN	SVM	GISR	GISW	M-Model		IDEAL	
acq	0.829	0.870	0.859	0.931	0.932	0.909	0.932	GISR	0.932	GISR
corn	0.614	0.867	0.690	0.832	0.867	0.885	0.885	GISW	0.885	GISW
crude	0.793	0.853	0.823	0.871	0.813	0.869	0.869	GISW	0.871	SVM
earn	0.956	0.969	0.956	0.980	0.959	0.962	0.980	SVM	0.980	SVM
grain	0.803	0.887	0.820	0.917	0.804	0.910	0.910	GISW	0.917	SVM
interest	0.481	0.881	0.721	0.962	0.721	0.881	0.881	GISW	0.962	SVM
money-fx	0.582	0.718	0.674	0.717	0.681	0.756	0.756	GISW	0.756	GISW
ship	0.800	0.860	0.800	0.845	0.825	0.872	0.860	WH	0.872	GISW
trade	0.732	0.763	0.740	0.715	0.714	0.788	0.788	GISW	0.788	GISW
wheat	0.713	0.839	0.727	0.820	0.825	0.875	0.875	GISW	0.875	GISW
Top 10 Macro BE	0.730	0.851	0.781	0.859	0.814	0.871	0.874		0.884	

Table 3. Macro recall and precision break-even point measures of the 10 most frequent categories, for individual classifiers, meta-model approach (M-Model) and the ideal combination (IDEAL).

Since the ideal combination always consists of the best algorithm for each category, it sets an upper bound for the amount of improvement that can be made under our meta-model. Table 4 shows the comparison of performances, under different groups of categories, between meta-model (M-Model) and the ideal combination (IDEAL). Based on the utility measures as shown in the table, we will look into how much improvement the meta-model (M+(%)) has achieved within the improvement bound (I+(%)) set by the ideal combination in Table 5.

Utility Measure	M-Model	IDEAL
Top 10 Macro BE	0.874	0.884
All 90 Micro BE	0.858	0.868
All 90 Macro BE	0.656	0.692
Remaining 80 Macro BE	0.628	0.668

Table 4. Classification performances of meta-model (M-Model) and the ideal combination (IDEAL) under different groups of categories.

In Table 5, among all other algorithms, the classification improvement made by either meta-model (M+(%)) or the ideal combination (I+(%)) over Rocchio is

the largest, more than 10%, in all aspects. Apart from improvement for frequent categories, the meta-model can also achieve improvement for other rare categories (Remaining 80 Macro BE). The table also shows that the improvement made by meta-model over individual component algorithms is quite impressive, when considering the improvement bound set by the ideal combination (I+(%)). Improvement achieved by meta-model within the improvement bound of the ideal combination ($M + /I + (%)$) is also presented in the table. As for Top 10 Macro BE, when compared with Rocchio and k-NN, the meta-model has attained more than 90% of the improvement bound.

	RO	M+(%)	I+(%)	M+/I+(%)	WH	M+(%)	I+(%)	M+/I+(%)
Top 10 Macro BE	0.730	19.622	21.019	93.355	0.851	2.692	3.891	69.184
All 90 Micro BE	0.776	10.567	11.856	89.130	0.820	4.634	5.854	79.167
All 90 Macro BE	0.578	13.495	19.723	68.421	0.649	1.079	6.626	16.279
Remaining 80 Macro BE	0.559	12.421	19.427	63.939	0.623	0.822	7.105	11.572
	KNN	M+(%)	I+(%)	M+/I+(%)	SVM	M+(%)	I+(%)	M+/I+(%)
Top 10 Macro BE	0.781	11.857	13.163	90.078	0.859	1.700	2.887	58.871
All 90 Micro BE	0.802	6.983	8.229	84.848	0.841	2.021	3.210	62.963
All 90 Macro BE	0.607	8.072	14.003	57.647	0.640	2.500	8.125	30.769
Remaining 80 Macro BE	0.585	7.429	14.124	52.602	0.613	2.512	8.900	28.225
	GISR	M+(%)	I+(%)	M+/I+(%)	GISW	M+(%)	I+(%)	M+/I+(%)
Top 10 Macro BE	0.814	7.309	8.562	85.366	0.871	0.333	1.505	22.137
All 90 Micro BE	0.830	3.373	4.578	73.684	0.845	1.538	2.722	56.522
All 90 Macro BE	0.625	4.960	10.720	46.269	0.655	0.153	5.649	2.703
Remaining 80 Macro BE	0.602	4.435	10.943	40.528	0.628	0.018	6.250	0.286

Table 5. Improvement of classification performances of meta-model ($M+(%)$) and the ideal combination ($I+(%)$) over individual algorithms, and improvement achieved by meta-model within the improvement bound set by the ideal combination ($M+/I+(%)$).

Algorithm Combination	All 90 Micro BE	All 90 Macro BE	Top 10	Remaining 80
1) KNN+WH+RO	0.820	0.649	0.851	0.623
2) GIS-R+KNN+WH+RO	0.842	0.650	0.852	0.624
3) GIS-W+KNN+WH+RO	0.848	0.657	0.870	0.631
4) GIS-W+SVM+KNN+WH+RO	0.854	0.657	0.871	0.630
5) GIS-W+GIS-R+SVM+KNN+WH+RO	0.858	0.656	0.874	0.628

Table 6. Performance with different combinations of classifiers based on our meta-model under different groups of categories.

Table 6 shows that incremental improvement can be obtained as more robust and more classifiers are included in the algorithm pool of the meta-model. Performance obtained after adding GIS-R to Combination 1 is increased under different groups of categories. After replacing GIS-W with GIS-R, the improvement over Combination 1 is more significant. After adding the robust SVM to Combination 3, performance, measured by micro-averaged recall and precision break-even point measures (All 90 Micro BE), is further increased, as indicated in Combination 4. Combination 5 is actually the whole algorithm pool of our meta-model. The results

demonstrate that our meta-model is not limited to combining a fixed number of classifiers, or combining classifiers of same type, instead, it allows flexible additions or substitutions of different classifiers in its algorithm pool.

7 Conclusions

We have developed a similarity-based textual document categorization method called the generalized instance set (GIS) algorithm. GIS integrates the advantages of linear classifiers and k-nearest neighbour algorithm by generalization of selected instances. Experimental results show that our new approach outperforms the k-NN approach, linear classifiers and the latest SVM in most of the experiments. To further enhance the performance, we have proposed a meta-model framework which combines the strength of different variants of GIS algorithm as well as state-of-the-art existing algorithms using multivariate regression analysis on document feature characteristics. Different from existing categorization methods, our proposed meta-model can automatically recommend a suitable algorithm for each category based on the category-specific statistical characteristics. We have demonstrated our contributions including the application of meta-model approach on text categorization, resulting in enhanced classification performance, and the concrete experimental results to support our approach that our meta-model shows a better performance than existing algorithms such as k-NN, SVM and linear classifiers, namely WH and Rocchio.

Bibliography

- [1] C. Apte, F. Damerau, and S. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, 1993.
- [2] P. K. Chan and S. J. Stolfo. Comparative evaluation of voting and meta-learning on partitioned data. In *Proceedings of the International Conference on Machine Learning (ICML'95)*, pages 90–98, 1995.
- [3] W. W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. In *Proceedings of the Nineteenth International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 307–315, 1996.
- [4] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the Seventh International Conference on Information and Knowledge Management*, pages 148–155, 1998.
- [5] D. A. Hull, J. O. Pedersen, and H. Schutze. Method combination for document filtering. In *Proceedings of the Nineteenth International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 279–287, 1996.
- [6] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European Conference on Machine Learning (ECML'98)*, pages 137–142, 1998.
- [7] J. J. R. Jr. *Relevance feedback in information retrieval*. The SMART Retrieval System: Experiments in Automatic Document Processing, editor: Gerard Salton, Prentice-Hall, Inc, Englewood Cliffs, News Jersey, 1971.
- [8] W. Lam, K. F. Low, and C. Y. Ho. Using a bayesian network induction approach for text categorization. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence, (IJCAI), Nagoya, Japan*, pages 745–750, 1997.

- [9] L. S. Larkey and W. B. Croft. Combining classifiers in text categorization. In *Proceedings of the Nineteenth International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 289–297, 1996.
- [10] D. D. Lewis, R. E. Schapore, J. P. Call, and R. Papka. Training algorithms for linear text classifiers. In *Proceedings of the Nineteenth International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 298–306, 1996.
- [11] K. M. Ting and I. H. Witten. Stacked generalization: when does it work? In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pages 866–871, 1997.
- [12] V. Vapnic. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.
- [13] B. Widrow and S. Stearns. *Adaptive Signal Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1985.
- [14] Y. Yang. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In *Proceedings of the Seventeenth International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 13–22, 1994.
- [15] Y. Yang, T. Ault, and T. Pierce. Combining multiple learning strategies for effective cross validation. In *Proceedings of the International Conference on Machine Learning (ICML 2000)*, pages 1167–1174, 2000.
- [16] Y. Yang and C. D. Chute. An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems*, 12(3):252–277, 1994.
- [17] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the Twenty-First International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–49, 1999.
- [18] Y. Yang and J. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the International Conference on Machine Learning (ICML'97)*, pages 412–420, 1997.