TTI's System for the LSHTC3 Challenge

Yutaka Sasaki and Davy Weissenbacher

Toyota Technological Institute
2-12-1 Hisakata, Tempaku-ku, Nagoya 468-8511, Japan
{Yutaka.Sasaki,Davy.Weissenbacher}@toyota-ti.ac.jp

Abstract. In this paper, we describe TTI's system for the third edition of the PASCAL Challenge on Large-Scale Hierarchical Text Classification (LSHTC3). We participated in the Wikipedia Medium subtask (Track 1) in which a system is expected to classify 81,262 test documents into a class hierarchy of 50,312 categories (36,504 leaf categories) based on 456,866 training documents. We used unigram-based input vectors of the training documents which are provided by the challenge organizers. We took a supervised approach using Support Vector Machines (SVMs). Our system fully utilizes the hierarchical structure of classes in the top-down manner. The features of our system include (1) SVM edge classification, (2) SVM thresholding adjustment, and (3) global pruning. In our system, each edge, *rather than node*, of the class hierarchy is associated with a binary SVM classifier. In the top-down classification, edge SVM classifiers decide whether data should be passed from the parent node to the child node. The SVM decision bias has been adjusted to improve the recall. Our system also globally prune unlikely assignments of classes after gathering classification results over the hierarchy. In the formal evaluation, our system is the second, and third within 17 systems with respect to the hierarchical F1, and other ranking metricss, respectively. Our system's characteristics is that it tends to predict minor categories well due to our three system features, which is indicated by the best macro-average F1 score among participants.

1 Introduction

Text Classification has been activity studied for labeling natural language texts with thematic categories from a predefined set. [17] Thematic categories are also called *topics* (or *genres*). Support Vector Machines (SVMs) [19, 2] have been successfully applied to Text Classification since Dumais et al. [3].

In this paper, we employ a fully supervised approach where we assume one-to-one correspondence between categories (in text classification context) and classes (in the machine learning context). We frequently face a computational challenge in large-scale text classification problems. We need to handle tens of thousands classes to classify millions of documents. For example, in the biomedical field, we need to classify biomedical documents into the ICD-9 or ICD-10 codes, which consist of more than 150,000 medical codes. Moreover, ICD codes are hierarchically structured. The CMC Medical NLP Challenge 2007 [13], which was organized by the Computational Medicine Center (CMC), targeted the task of assigning ICD-9-CM clinical codes; however, the number of categories was limited to 45. MEDLINE abstracts reached nearly 20 million which should be classified into Medical Subject Headings (MeSH)[10] terms, which consist of more than 110,000 medical concepts. In this respect, we have been studying large-scale text classification where classes are hierarchically structured.

Automatically assigning Wikipedia hierarchical categories to a document is a typical multi-topic text classification problem. The third edition of PASCAL Challenge on Large-Scale Hierarchical Text Classification [11] includes the task of assigning Wikipedia categories to documents.

In this paper, we present a hierarchically ordered SVMs and demonstrate how to achieve a high performance in the large-scale hierarchical text classification.

2 PASCAL LSHTC3 Task Overview

The third PASCAL Challenge on Large-Scale Text Classification (LSHTC3) provided three kinds of tasks.

Track 1 Large Scale Hierarchical Classification.
- Wikipedia Medium
- Wikipedia Large

Track 2 Multi-task Learning
Track 3 Refinement Learning

We participated in the Wikipedia Medium subtask in Track 1. This track is a large-scale hierarchical classification task based on Wikipedia data. The subtask is to evaluate hierarchical text classification technologies on a medium-sized Wikipedia data set. Participants in this subtask are expected to classify 81,262 test documents into a hierarchy of 50,312 categories (36,504 leaf categories) based on 456,866 training documents. This is computationally challenging which leads to algorithmic and theoretical challenges in Computer Science. Only leaf categories are assigned to the challenge data set. The participants were provided with the pre-processed version of the data set as well as original documents. The class hierarchy for this subtask is a Directed Acyclic Graph (DAG) with the root node.

3 TTI's System: Training Stage

3.1 Bottom-up Propagation

Since only leaf categories are assigned to data, first, we need to propagate training samples from leaves to the root of the class hierarchy so that classes are assigned not only to leaves but also to intermediate and top nodes, based on the transitivity of inclusion relations between Wikipedia categories.

Figure 1 illustrates propagation of documents in a hierarchy consisting of six categories A-F. In this figure, sample x1 is assigned to categories D and E,

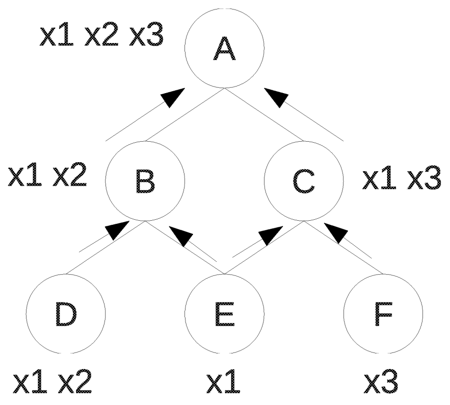

Fig. 1. Bottom-up propagation of training data

sample x2 is assigned to D, and sample x3 is assigned to F. Let us look at the case of x1 assigned to E. x1 of E is propagated to both categories B and C. Then, x1 of B is propagated to A. When x1 is propagated from C to A afterward, to avoid redundant propagation, the propagation of x1 (originally from E via C) terminates at A, even if A had a parent category.

To perform the propagation of the sample, we employ a recursive algorithm. The algorithm is described in pseudo-code in Algorithm 1. A sample is propagated in a directed acyclic graph in the bottom-up manner.

Algorithm 1 propagate(*node*, *sample*)

```
1. If sample is already assigned to node, return.
2. Assign sample to node.
3. If node is the root, return.
4. Foreach n in parents of node do
     – call propagate(n,sample).
```

3.2 Top-down training

This section describes the top-down training. Based on the propagation of training samples in the hierarchy, we train SVM classifiers for each edge of the hierarchy to estimate how strong is the relation of the sample with the child nodes of the current node. In other words, our SVM classifiers represent edges, rather than nodes, in the hierarchy. Each edge is coupled with a binary-class SVM using one-against-the-rest approach [20].

Figure 2 illustrates the training of SVM classifiers based on the assignments of samples to hierarchical categories (shown in Fig.1). Let us look at the training at node B. As x1 and x2 are assigned to node B, here we consider only x1 and

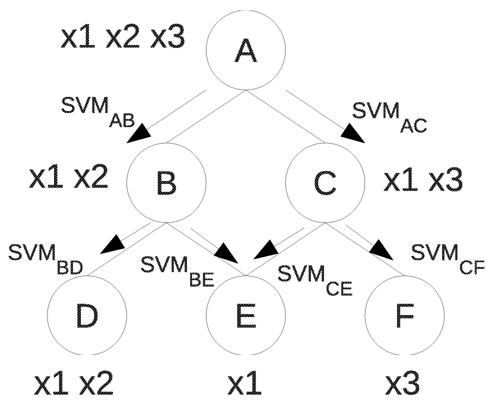

Fig. 2. Top-down training

x2. SVM$_{BD}$ is trained so as to classify both x1 and x2 to D whereas SVM$_{BE}$ is trained so as to classify x1 to E but not x2 to E.

For example, SVM$_{BE}$ is trained on the binary classification data

$$\{(x1, +1), (x2, -1)\}.$$

Training depending on local branches is beneficial in restricting the number of samples. It is also effective to ease positive-negative data imbalance.

Another benefit of edge-oriented SVM classification is that a classifier can capture a local characteristics of data distribution. Naturally, x1 classified into E from nodes B and C should need a different consideration. The former is classification based on x1 and x2 and the latter is based on x1 and x3. This indicates that SVM$_{BE}$ and SVM$_{CE}$ are different models.

Algorithm 2 shows the top-down training of edge-oriented SVM classifiers.

Algorithm 2 train_node(*node*)

```
1. If node is already explored, return.
2. If node is a leaf, return.
3. Let $X$ be the set of samples assigned to node.
4. Foreach $n$ in child nodes of node do
     – Foreach $x$ in $X$ do
         • Set label +1 to $x$ if $x$ is assigned to $n$; otherwise set label -1 to $x$;
     – Train SVM$_{node,n}$ on the generated data.
     – call train_node($n$).
```

4 TTI's System: Classification Stage

The classification of (unlabeled) test data is also conducted in the top-down manner.

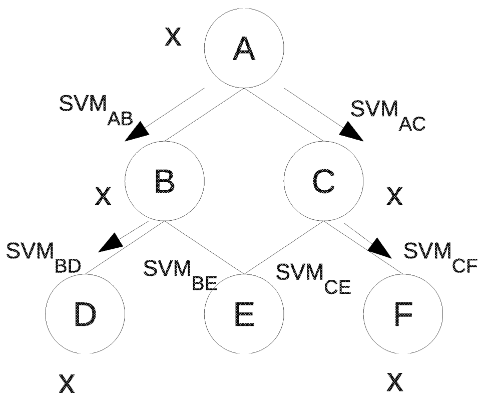

Fig. 3. Top-down classification

4.1 Top-down Classification and Confidence Values

Figure 3 illustrates top-down classification of test data x. First, x is classified to B and C, based on the decision by SVM$_{AB}(x)$ and SVM$_{AC}(x)$, respectively. To adjust the effect of positive-negative sample imbalance, we set a bias β. When SVM$_{pc}(x) > \beta$, x is classified from parent category p to child category c. When both SVM$_{AB}(x) > \beta$ and SVM$_{AC}(x) > \beta$ are satisfied, x are classified into both A and B.

Along with the hierarchical classification, we keep track of confidence scores of the classification. The output value of SVM(x) is converted to [0,1] range using the sigmoid function:

$$\sigma(x) = \frac{1}{1 + \exp(-x)}.$$

When x reaches a leaf node n, the confidence score $c(x, n)$ is calculated as follows:

$$c(x, n) = \prod_{(n_1,n_2) in E} \sigma(SVM_{n_1 n_2}(x)),$$

where E is the set of edges that x has followed in the path from the root to leaf n.

Algorithm 3 shows the classification algorithm. The classification procedure is initiated with classify(x,root,1).

Algorithm 3 classify(x,*node*,*confidence*)

1. If *node* is a leaf, assign x to *node*. If x is already assigned to *node*, set $c(x, node)$ as max($confidence$,old $c(x, node)$), otherwise set $c(x, node) = confidence$.

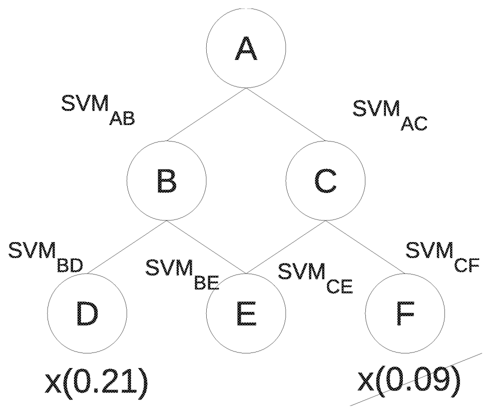

Fig. 4. Global pruning

2. Foreach n in child nodes of *node* do
   – If $\text{SVM}_{node,n}(x) > \beta$, call classify$(x$,$n$,confidence*$\sigma(\text{SVM}_{node,n}(x)))$.
3. If *none* of child nodes satisfies $\text{SVM}_{node,n}(x) > \beta$, choose

$$n' = \underset{n}{\text{argmax}}\, SVM_{node,n}(x).$$

Then call classify$(x$,n',confidence*$\sigma(\text{SVM}_{node,n'}(x)))$.

The last step of Algorithm 3 is important. When there is no child class in which x is clearly to be classified, we choose a child node that has the best SVM classification value for x. This is of important because minor classes tend to have a small or negative SVM classification value. This step improves recall of the classification.

4.2 Global Pruning

After the classification of x throughout the hierarchy, we prune unlikely classes for x. We set the global threshold θ. When there are multiple nodes are assigned to x, if c$(x$,$n)$< θ, the assignment of x to n is removed.

Figure 4 illustrates the global pruning. In Fig. 4, x is classified into D and F. Here, the confidence score of x in D is 0.21 and in F is 0.09. When $\theta = 0.3$, both of the confidence scores are below the threshold. However, we need to choose at least one category for every test data point. Therefore, only assignment of x to F is removed.

5 Results

5.1 Experimental Settings

We have created 65,333 SVM classifiers that correspond to edges in the Wikipedia medium-size category hierarchy. We used SVM^{perf} [5, 6] with the linear kernel.

Table 1. Results

C	β	θ	Acc	EBF	LBMaF	LBMiF	HF
1.0	-0.2	0.00	0.3603	0.4316	0.2763	0.3833	0.6197
1.0	-0.2	0.10	0.4140	0.4742	0.2889	0.4652	0.6875
1.0	-0.2	0.11	0.4172	0.4758	0.2860	0.4692	0.6901
1.0	**-0.2**	**0.12**	**0.4200**	**0.4771**	**0.2835**	**0.4725**	**0.6922**
1.0	-0.2	0.13	0.4220	0.4778	0.2803	0.4751	0.6939
1.0	-0.2	0.14	0.4236	0.4781	0.2767	0.4767	0.6951
1.0	-0.2	0.15	0.4244	0.4778	0.2730	0.4775	0.6956
1.0	-0.2	0.16	0.4245	0.4770	0.2689	0.4776	0.6959
1.0	-0.2	0.17	0.4240	0.4759	0.2645	0.4769	0.6957

The SVM parameter C is set to 1.0 in the traditional SVM sense. For SVM^{perf}, an actual value $C^{perf} = N * C/100$, where N is the number of samples in each node. Note that C^{perf} varies in each node in the top-down training.

The feature values in the given data set represent the number of occurrences of each unigram. To avoid excessive effects of large feature values, we normalized the feature value with the function $v/(v+1)$ where v is the original feature value.

5.2 Evaluation Metrics

Evaluation metrics defined in LSHTC3 are as follows:

- Accuracy (Acc)
- Example-based F1 measure (EBF)
- Label-based Macro-average F1 measure (LBMaF)
- Label-based Micro-average F1 measure (LBMiF)
- Hierarchical F1-measure (HF) [7]

5.3 Processing Time

We assessed the training and classification time on the Core i7 3.0GHz PC with 8GB memory. The total training time on 456,886 data was 58,389s, which is about 16hr. In the test stage, 81,262 data has been classified in 9,864 (=2.7hr).

5.4 Scores

Table 1 shows our results in the formal evaluation. We set $\beta = -0.2$ and varied $\theta \in \{0.00, 0.10, 0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17\}$. We submitted our result when C=1.0, $\beta = -0.2$, $\theta = 0.12$ as the macro-F1 and micro-F1 is balanced best (i.e., we chose the result with the maximum sum of the two F scores).

6 Discussions and Related Work

Text classification is a typical multi-class multi-label classification problem. To efficiently solve multi-class classification, Platt [14] proposed DAG SVMs that utilize a DAG structure to accelerate multi-class classification; however, it is not classification on a given hierarchy. Studies to solve multi-class multi-label classification problems have been summarized in [18].

There have been many studies that use local context in hierarchical classification. [8, 1, 12, 16] Chakrabarti et at. [1] proposed a naive-Bayes document classification system that follows hierarchy edges from the root node. Koller and Sahami [8] applied Bayesian networks to hierarchical document classification. In our approach, we applied SVMs which have shown good classification performance in exchange of expensive training cost. Our approach was challenging as the size of the data and hierarchy is much larger than previous studies. We also devised an effective pruning method and confirmed its advantage in the experiments.

Large-scale hierarchical classification is a further challenging problem. A wide variety of approach have been taken to handle hierarchical classes in text classification. Among the approaches proposed during the previous LSHTC challenges, Malik [9] evaluated SVM Hierarchical and SVM Flat in the LSHTC1 Challenge. Comparing several cases of flattening of a hierarchy. It shows that flattening of a hierarchy reduces compound errors during the traversal of hierarchy. In our system, we tried to get a more benefit from the hierarchical structure than the disadvantage in accumulating errors. In LSHTC2, the best accuracy on Wikipedia Small (= Wikipedia Medium in LSHTC3) was 37.39% whereas our accuracy reached 42.00%. This implies that our approach has been successful in making the most of the hierarchical structure.

Impact of the Bias

Iannou et al. [4] summarizes thresholding methods in multi-label classification. Basically, bias β is a threshold that adjusts PCut [21]. Note that SVMs automatically set the threshold b in the SVM decision function $SVM_{pc}(x) = w_{pc} \cdot x + b_{pc}$. Setting β means (adhoc) threshold adjustment $SVM_{pc}(x) = w_{pc} \cdot x + b_{pc} + \beta$, where p and c are parent and child categories, respectively. From Table 2, we can see how bias β affects the scores.

Impact of the Global Pruning

From Table 1, we can see that the global pruning is quite effective. The scores without the global pruning (*i.e.*, $\theta = 0$) drop by about 5% from the best scores. Punera and Rajan[15] proposed a regularized unimodal regression to smooth classification scores in a hierarchy. Following this study, we want to find a better way to improve our calculation of confidence scores so as to prune unlikely category assignments more accurately.

Table 2. Results without bias β

C	β	θ	Acc	EBF	LBMaF	LBMiF	HF
1.0	0.0	0.00	0.3850	0.4485	0.2734	0.4241	0.6504
1.0	0.0	0.10	0.4181	0.4728	0.2776	0.4700	0.6909
1.0	0.0	0.11	0.4201	0.4738	0.2755	0.4723	0.6926
1.0	0.0	0.12	0.4221	0.4747	0.2736	0.4742	0.6940
1.0	0.0	0.13	0.4235	0.4751	0.2713	0.4757	0.6951
1.0	0.0	0.14	0.4243	0.4751	0.2682	0.4762	0.6957
1.0	0.0	0.15	0.4246	0.4747	0.2652	0.4763	0.6959
1.0	0.0	0.16	0.4242	0.4736	0.2617	0.4756	0.6958
1.0	0.0	0.17	0.4233	0.4724	0.2578	0.4744	0.6953

Impact of SVM Parameter C

The SVM parameter C affects the performance. For $\beta = -0.2, \theta = 0.12$, the HF scores of $C = 1.0$, $C = 0.5$, and $C = 0.1$ have been 0.6922, 0.6951, and 0.6945,respectively. The value of SVM parameter C can be decided by cross validation but it is a time consuming process. As C affects classification performance, this remains an issue to improve our system results.

Normalization of Confidence Scores

We attempted to normalize confidence scores so that the confidence scores $c(x, n)$ can be regarded as probabilities; however, the normalization slightly degraded the scores. This can be explained as follows. Without normalization, confidence scores based on one-vs-the-rest classifiers are independent each other in a node and reflect the distribution of training data but after normalization, the confidence scores become relative values, affected by test data in a node, whilst the number of test data is much smaller than that of the training data.

Development Set Results

After the formal evaluation, we tried to confirm that the parameters that we used could have been selected using a development data set. We leave 1/10 of the training data for test and used 9/10 of the training data for training the SVM top-down classifiers.

Figure 5 shows the Micro-F1 scores according to the different θ. In this experiment, $C = 1.0$ and $\beta = -0.2$. The peak of the curve is at $\theta = 0.15$. This is a consistent result to the formal evaluation result (Table 1). This indicates that it is possible to optimize the parameters to achieve better performance without depending on the LSHTC evaluation site.

Table 3. Evaluation Results in the Wikipedia Medium subtask

Name	Acc	EBF	LBMaF	LBMiF	HF
arthur	0.4382	0.4937	0.2674	0.4939	0.7092
coolvegpuff	0.4291	0.4824	0.2507	0.4779	0.6892
TTI	0.4200	0.4771	0.2835	0.4725	0.6922
chrishan	0.4117	0.4768	0.2454	0.4187	0.6766
anttip	0.4077	0.4460	0.2385	0.4309	0.6803
dhlee	0.3848	0.4352	0.2824	0.4209	0.6662
szarak	0.3711	0.4366	0.2195	0.3769	0.6447
brouardc	0.3536	0.4182	0.2398	0.3739	0.6428
daq	0.3526	0.3896	0.1929	0.3653	0.6330
SSir	0.3270	0.3873	0.1681	0.3879	0.6394
hautcs	0.3204	0.3473	0.1037	0.3274	0.6030
KULeuven	0.2976	0.3408	0.1825	0.3045	0.5494
Peaceguard	0.2499	0.2917	0.0555	0.2754	0.5916
Knn Baseline	0.2491	0.3176	0.1758	0.2979	0.5609
TUD_KE	0.2448	0.2839	0.1303	0.2762	0.5368
dicaro	0.0626	0.0805	0.0211	0.0716	0.3448
glouppe	0.0473	0.0846	0.1758	0.0971	0.6676

Table 4. Additional Best Result (unofficial)

C	β	θ	Acc	EBF	LBMaF	LBMiF	HF
0.5	-0.2	0.14	0.4289	0.4827	0.2710	0.4828	0.6995

Additional Evaluations

In the additional evaluations after the formal evaluation, our parameter improvement leads to the second best hierarchical F1 (HF) and micro-average F1 (LBMiF) scores (Table 4). The accuracy is almost equivalent to the second best whereas the macro-average F1 is the highest among the top three systems (Table 3).

Computational Complexity

Assuming SVM is linear to the number of data, the computation complexity of our top-down training is $O(kn \log m)$ and that of our top-down classification is $O(kn \log m)$, where n is the number of data, k is an average number of classes per data point, and m is the number of leaf categories.

7 Conclusions

We have successfully created a large-scale hierarchical text classification system. Our system has the following features:

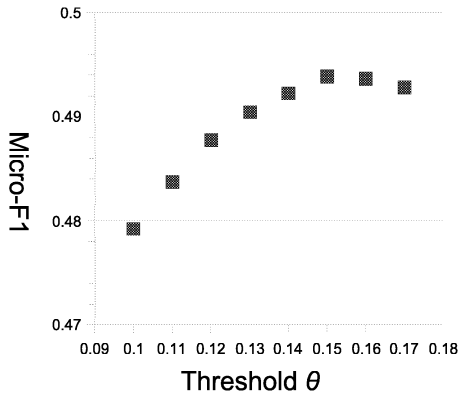

Fig. 5. Micro-F1 scores vs θ on the development set

- Bottom-up propagation of examples;
- Top-down training of SVM edge classifiers;
- Top-down classification with the threshold adjustment;
- Global pruning.

In the formal evaluation, our system is ranked first, second, and third within 17 systems with respect to the macro-average F, the hierarchical F, and other measures, respectively.

While our system exhibited good performance in Wikipedia Medium subtask, our approach is not efficient enough to deal with Wikipedia Large data sets. Our future work includes the development of much more efficient hierarchical SVM algorithms, their acceleration using GPGPUs, and the improvement of the confidence score and global pruning method.

References

1. S. Chakrabarti, B. Dom, R. Agrawal, and P. Raghavan.: Scalable feature selection, classification and signature generation for organizing large text databases into hierarchical topic taxonomies. International Journal on Very Large Data Bases 7(3), pp163–178 (1998)
2. Cortes, C., Vapnik, V.M.: Support Vector Networks. *Machine Learning*, Vol. 20, pp. 273–297 (1995)
3. Dumais, S.T., Platt, J., Heckerman, D. , Saham, M.: Inductive Learning Algorithms and Representations for Text Categorization. *Prof. CIKM '98*, pp.148–155 (1998)
4. Ioannou, M., Sakkas, G., Tsoumakas, G., Vlahavas, I.: Obtaining bipartitions from score vectors for multi-label classification. *In Proc. of IEEE International Conference on Tools with Artificial Intelligence*, pp. 409–416 (2010)
5. Joachims, T.: A Support Vector Method for Multivariate Performance Measures, *Proc. of the International Conference on Machine Learning (ICML-05)*, pp. 377–384 (2005)

6. Joachims, T.: Training Linear SVMs in Linear Time. *Proc. of the ACM Conference on Knowledge Discovery and Data Mining (KDD-06)*, pp. 217–226 (2006)
7. Kiritchenko, S.: Hierarchical text categorization, its application to bio-informatics. *Ph.D. thesis*, University of Ottawa Ottawa, Ont., Canada (2006)
8. D. Koller and M. Sahami: Hierarchically classifying documents using very few words. In Proc. of the Fourteenth International Conference on Machine Learning (97), pages 170–178 (1997)
9. Malik, H.: Improving hierarchical svms by hierarchy flattening and lazy classification. *In Proc. of the ECIR 2010 Large Scale Hierarchical Classification Workshop* (2010)
10. Medical Subject Headings, http://www.nlm.nih.gov/mesh/
11. PASCAL LSHTC3 Challenge, http://lshtc.iit.demokritos.gr/.
12. A. McCallum, R. Rosenfeld, T. Mitchell, and A. Ng: Improving text classification by shrinkage in a hierarchy of classes. In Proc. of the Fifteenth International Conference on Machine Learning, pp. 359–367 (1998)
13. Pestian, J.P., Brew, C., Matykiewiczk, P., Hovermale, D.J., Johnson, N., K.ulti-label Classification of Clinical Free Text. *Proc of ACL-2007 Workshop on BioNLP*, pp. 97–104 (2007)
14. Platt, J.: Fast Training of Support Vector Machines using Sequential Minimal Optimization. *Advances in Kernel Methods - Support Vector Learning*, MIT Press (1998)
15. Punera, K., Rajan, S.: Improved Multi Label Classification in Hierarchical Taxonomies, *In Proc. of IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 388–393 (2009)
16. Susan T. Dumais and Hao Chen. Hierarchical classification of Web content. Proc. of 23rd ACM International Conference on Research and Development in Information Retrieval, pp. 256–263, Athens, (2000)
17. Sebastiani, F.: Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, Vol. 34, No.1, pp.1–47 (2002)
18. Tsoumakas, G., Katakis, I.: Multi-Label Classification: An Overview. *International Journal of Data Warehousing and Mining*, Idea Group Publishing, 3(3), pp. 1-13 (2007).
19. Vapnik, V.: *The Nature of Statistical Learning Theory*, Springer (1995).
20. Weston, J., Watkins, C.: Support Vector Machines for Multiclass Pattern Recognition. *Proc. of the Seventh European Symposium on Artificial Neural Networks*, Brussels, pp. 219-224(1999)
21. Yang, Y.: A study on thresholding strategies for text categorization. *in Proc. of the 24th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR-2001)*, New York, NY, USA, pp. 137–145 (2001)