The ECIR 2010 Large Scale Hierarchical Classification Workshop

A. Kosmopoulos[1 2], E. Gaussier[3], G. Paliouras[1], S. Aseervatham[3]
[1]National Center for Scientific Research "Demokritos"
[2]Athens University of Economics and Business, Greece
[3]Lab. d'Informatique de Grenoble & Grenoble University, France
{*akosmo, paliourg*}*@iit.demokritos.gr*
{*eric.gaussier, Sujeevan.Aseervatham*}*@imag.fr*

May 2, 2010

Abstract

This paper reports on the *Large Scale Hierarchical Classification* workshop (http://kmi.open.ac.uk/events/ecir2010/workshops-tutorials), held in conjunction with the European Conference on Information Retrieval (ECIR) 2010. The workshop was associated with the PASCAL 2 Large-Scale Hierarchical Text Classification Challenge (http://lshtc.iit.demokritos.gr), which took place in 2009. We first provide information about the challenge, presenting the data used, the tasks and the evaluation measures and then we provide an overview of the approaches proposed by the participants of the workshop, together with a summary of the results of the challenge.

1 Introduction

Hierarchies are becoming ever more popular for the organization of documents, particularly on the Web (Web directories are an example of such hierarchies). Along with their widespread use comes the need for automated classification of new documents to the categories in the hierarchy. Large-scale hierarchical classification, i.e. the problem of classifying objects in a hierarchical category system comprising several thousand of categories or more, has been investigated for more than a decade now. The seminal work of [22] on Ohsumed, involved ca. 14,000. Similarly, the data used in the TREC-10 Filtering task in 2001 involved ca. 5,000 categories. More recently, the study reported initially in [20] and then in [9, 10], aimed at assessing the behaviour of several categorization methods in the context of large category systems.

Since then, there has been a continuous interest in the subject as well as a continuous flow of new ideas. In recent work, such as in [19], the dichotomy between flat and hierarchical approaches is somewhat blurred and the question on which approach to use remains open. Furthermore, in the different studies on the subject, several classifiers are used differently, e.g. with different pre-processing chains and/or sampling strategies. Thus a direct comparison of the methods proposed is in general not possible. Lastly, recent challenges on large-scale classification focused on situations involving a large number of high-dimensional examples, but few categories.

All these reasons led us to propose a challenge on large-scale classification which, in addition to involving a large number of high-dimensional examples, would also involve a large hierarchy of categories. Based on the results of the challenge, we organized a workshop, in order to discuss the methods used in the challenge, as well as those proposed in independent studies.

The remainder of the paper is organized as follows: Section 2 provides an overview of the challenge; Section 3 then gives an overview of the different approaches used by the workshop participants, whereas Section 4 provides a summary of the results obtained; Section 5 is finally deoted to the conclusion.

2 Presentation of the Challenge

The large Scale Hierarchical Text classification (LSHTC) PASCAL 2 Challenge attracted the interest of the research community by offering an opportunity to try on new ideas on this hard problem. The state-of -the-art methods in the area participated in the competition, setting a standard against which new methods can compare. The new LSHTC is available as a benchmark for further experiments. While an oracle is maintained at the website[1] of the competition and can be used to obtain an objective estimate of the performance of new methods.

The launch of the challenge was officially announced on the 10^{th} of the July of 2009. On that date the datasets were made publicly available at the website of the challenge, along with the details regarding the tasks, rules and guidelines. Until the 27^{th} of November the participants were able to upload their results in each task and decide on whether to publish the or not. At the end of this period the submission system was locked. Visitors were still able to use the evaluation system as an oracle, but without the option of publishing the results. Following the competition, the participants were given the chance to submit executables of their systems, which were used to assess the scalability of the methods. Furthermore, they were asked to submit short descriptions of their methods and experiments if they desired. These short papers were made available on the site of the competition.

Next we describe the datasets of the challenge and the procedure by which they were constructed. Then, we focus on the four tasks of the challenge, describing their nature and the motivation for their selection. At the end of this section we discuss the evaluation measures that were used.

2.1 Datasets and Data Preparation

The data used in the LSHTC challenge are a part of the ODP (Open Directory Project) directory. The documents we have retained have been indexed in two ways:

1. *Content vectors* are vectors obtained by directly indexing the web pages, using a standard indexing chain (pre-processing, stemming/lemmatization, stop-word removal)

[1] http://lshtc.iit.demokritos.gr/

2. *Description vectors* are vectors obtained by indexing the ODP descriptions of the web pages. The ODP descriptions are manually created by ODP editors when placing new documents into the ODP hierarchy. They are thus available for each document in the ODP hierarchy, but not for new documents.

In addition to the documents, all the categories of ODP were also coded as description vectors, using their ODP descriptions. In the part of the ODP considered, categories are organized as a tree in which only the leaves contain documents, the intermediate categories mainly serving maintenance and information access purposes.

Each of the indexed Web page belongs to a single category. Multi-labelled Web pages have not been used in the dataset. Two datasets were used during the challenge: a large one (12294 categories) and a smaller one (1139 categories). Each of the above datasets was randomly split into training, validation and test sets, per ODP category. Table 1 gives the number of documents in each of these sets. The participants had the chance to dry-run their classification methods on the small dataset and then they were asked to train and tune their system using the training and validation parts of the large set, providing their classification results on the test part.

Set	Large Dataset	Small Dataset
Training	93805	4463
Validation	34905	1860
Testing	34880	1858

Table 1: Number of documents used in each set

The LSHTC challenge consisted of four large-scale classification tasks, with partially overlapping data. These tasks, described below, emulated different scenarios for learning and using classifiers, and introduced different challenges for the machine learning community, such as large-scale datasets, hierarchical category systems and use of non independent and identical distributed data (i.i.d.) in training and testing. 19 participants took part in the first task, 5 in task 2 and 7 in tasks 3 and 4.

- Task 1: Basic

  In this task, participants were asked to train and test their system using content vectors only. This task corresponds to a standard text classification task in a large scale setting.

- Task 2: Cheap

  In this task, participants were asked to train their system using description vectors only; testing was then performed on content vectors. This is considered to be a “cheap” scenario because one does not need to crawl the pages in order to acquire training data. The word distributions in this task are different between training and testing (non i.i.d case).

- Task 3: Expensive

  In this task, participants were asked to train their system using both content and description vectors; as before, testing was performed on content vectors only. This is considered to be an “expensive” scenario because crawling of Web pages in required for training and also the description vectors are used. The motivation was the same as for the previous task, using this time all the available information for training.

- Task 4: Full

  In this task, participants were asked to train and test their system using both content and description vectors, i.e. all the information available. It is called “full” because it uses all the data that are available. The aim is to see how much the extra information affects the results.

2.2 Evaluation Measures

In order to measure classification performance we used an online evaluation system which is maintained on the website of the challenge. Participants submitted their results at a maximum frequency of once per hour per task. Once the results were submitted, the system measured the performance of the submission by computing the accuracy, macro-average F_1-measure and cumulative tree-induced error.

Accuracy is computed by the following formula:$Acc = \frac{\text{Correct Classifications}}{D}$, where D is the number of test documents. Macro-average F_1-measure is as usual $F_1 = \frac{2\cdot precision\cdot recall}{precision+recall}$, using macro-average versions of precision: $precision = \frac{\sum_{i=1}^{M} precision_i}{M}$ and recall: $recall = \frac{\sum_{i=1}^{M} recall_i}{M}$, where M is the number of categories. The tree-induced error [7] corresponds to: $Tree_error = \frac{\sum_{d=1}^{D} \text{Path-length}(c_d,t_d)}{D}$, where D is the number of testing documents, c_d the category in which document d was classified, and t_d the true category of the document.

Two statistical significant tests were used. The *p-test* was used for comparing proportions on the accuracy scores [21]: $p = \frac{p_a+p_b}{2}$ and $Z = \frac{p_a-p_b}{\sqrt{2p(1-p)/n}}$, where p_a and p_b are the two systems being compared and n is the number of trials, which in our case equals to the number of test vectors D. The macro sign test (*S-test*) was used for comparing the macro-average F_1 scores[21]: $Z = \frac{k-0.5n}{0.5\sqrt{n}}$, since $n > 12$, where n is the number of times a and b differ in terms of $F - 1$ score, k is the number of times that a achieves higher F_1 than b. In both tests the two systems are said to be significantly different if the P-value that corresponds to Z is less than 0.05.

Finally, as mentioned in section 1, participants were also asked to submit an executable of their system. Using the executable programs, we ran a scalability test, varying the size of the hierarchy from a hundred to a few thousand categories, in order to measure the computational (the time needed during training and classification) and memory (the maximum memory needed during training and classification) requirements of each method.

3 Overview of Approaches

Like most hierarchical category systems, intermediate categories in DMOZ mainly serve maintenance and navigation purposes, and do not constitute final categories, i.e. categories which are likely to receive documents. In such systems, final categories usually correspond to the leaves of the category tree. Xue *et al.* [19] propose to divide the approaches into two types: *big-bang* approaches, in which one tries to directly categorize documents into the leaf categories, without making use of the category hierarchy, and *top-down* approaches, in which the hierarchy is exploited to divide the overall classification problem into smaller ones. Studies reported in [9] and [12] suggest that the direct deployment of a single classifier onto the complete set of categories (big-bang approaches) may lead to training and classification time which are unacceptable. Quoting Liu *et al.* [9]: *"flat SVMs cannot be used in very large-scale real-world applications, due to their high computational complexity (an average response time, with 10 powerful machines running in parallel, of 0.69s for one single document is not acceptable for large-scale online classification)"*. On the contrary, top-down approaches, by simplifying the overall problem, lead to acceptable classification times. However they tend to propagate errors made at higher levels of the hierarchy. Participants to the challenge and/or workshop who have exploited the hierarchy have addressed this problem in two main ways, either by flattening the hierarchy, or by selecting in a first phase a part of the hierarchy. We will refer to the later type of approach as *mildly hierachical*.

3.1 Mildly Hierarchical Approaches

Flattening has been performed in the *arthur_general* system [18] in a very aggressive way, by retaining a two-level hierarchy consisting of the second level of the original hierarchy and all the leaves. The fact that the second level of the original hierarchy is used and not the first one is interesting, as this allows one to avoid the confusion that exists at the top of the hierarchy. The final hierarchy, consisting of two levels is simple and yet leads to a tractable problem as one first chooses one among a set of 311 categories, and then the final category among a limited set (each category has at most a few tens of daughters). A multi-class SVM classifier has been adopted to perform these selections.

Another approach to flattening is the one proposed in the *logicators* system [14], which replaces some categories by their daughters in order to obtain a desired depth k, where k parameter of the system. Here again, the obtained set of categories is organized hierarchically, and thus leads to reasonable classification times, but nevertheless uses few levels, and thus reduces the risk of making a mistake in the top-down process. In addition to this flattening, the author also proposes to maintain a set of possible categories at each level, in order to avoid making hard choices which cannot be revised. This is similar to the approach proposed in [8]. Once the set of possible leaf categories has been chosen by a hierarchical version of SVMs in a top-down fashion, a second multi-class classifier can be used to select the appropriate class among the leaves retained.

A third approach to exploit the hierarchy of categories, while simplifying it, is proposed by Xue *et al.* [19, 17] and used in the challenge. In this approach, a subset $\mathcal{S}_d$ of categories is first selected for a document d to be classified and then a specific classifier is trained for each subset. The subset $\mathcal{S}_d$ is determined on the basis of both the probability of categorizing d in $\mathcal{S}_d$ and the hierarchical structure. This selection is based on a standard k-NN classifier with a cosine similarity measure. The final classification relies on a Naïve Bayes classifier. This final classification step uses a subset of 10 leaf categories and can thus be performed efficiently once the Naïve Bayes classifier has been trained. One drawback of this approach is the requirement of training a document-specific classifier. Even though it is on a limited number of categories, this step slows down considerably the overall classification process.

Another interesting compromise between flat and hierarchical approaches is proposed in the *XipengQiu* system [15]. A centroid for each class is computed on the basis of both a standard term frequency representation of documents (averaged per category to obtain the centroid) and an IDF (inverse document frequency) representation for all the terms in the category. These two representations are linearly combined through a weight which needs to be tuned to the collection. Once the centroid of each cateogry c has been constructed (ω_c), the decision is made according to the distance of the document to be classified (d_t)

to all the centroids, taking into account the centroid of the parent category in the decision: $\text{argmax}_c < (\omega_c + \omega_{p(c)}), d_t >$, $p(c)$ representing the parent category of c and $<, >$ denoting the scalar product.

Lastly, an interesting approach, focusing on the input imbalance that typically occurs in large scale hierarchies, was proposed in [2]. The authors did not participate in the challenge but applied their method on a subset of the ODP category comprising 17 categories, with a depth of 5, and ca. 140,000 documents. The proposed method uses progressive filtering to unfold the hierarchy into a pipeline of binary classifiers. A threshold selection algorithm is then applied to recalibrate the binary classifiers, with the goal of minimizing the global error of a given pipeline. Recalibration associates to each binary classifier of a given pipeline a new threshold; If the score of the document to be categorized is above the threshold, it will be accepted and propagated down the pipeline. The results obtained show that progressive filtering outperforms a direct flat (big-bang) method.The system used in this particular setting is a specialization of the more general X.MAS system [1].

3.2 Big-bang Approaches

The *jhuang* system [13] proposes the use of an online training, flat classification approach. Two online algorithms were used for this purpose: (a) a variant of the OOZ algorithm proposed by the participating team in [11], and (b) a variant of the PA (passive-aggressive) algorithm proposed in [6]. In both cases, during the training phase, each time a new instance is misclassified, a weight matrix representing the importance of each feature is updated. Furthermore, to reduce the variance in prediction, a committee of models is learned by selecting the models obtained at different stages of the online process. As stated in section 1, one of the main disadvantages of the flat approaches is their computational cost in terms of training and classification. However, the authors report classification time for OOZ which is competitive to hierarchical approaches.

In the *NakaCristo* system [5], a simple k-NN algorithm is used with three changes. First, the weight given to features does not correspond to the standard tf-idf weight, but to the ConfWeight proposed in [16], which makes use of a parameter tuned here on the data sets. Second, the cosine similarity is transformed into a pseudo-distance which is used in the weighting rule of the kNN decision. Third, to increase the speed of the algorithm at classification time, the comparison is restricted to those documents which share at least a certain number of the most important features with the document to be classified.

In the *Brouard* system [4], a network is first constructed at training time linking terms and categories. A spreading activation function is then used during testing to determine the best categories for the documents according to the terms they contain. The system includes a number of parameters that need to be tuned to the collections. Furthermore, unlike other systems, the use of feature selection, through a simple IDF measure, is reported to improve the classification results.

Lastly, a very different approach was proposed in [3]. This system did not participate in the challenge. It adopts an approach based on error-correcting codes, developed on top of SVM classifiers. More precisely, the goal is to find coding matrices which satisfy different constraints desirable in the context of large scale text classification, like row and column dissimilarity and sparsity. The preliminary results obtained with the method seem encouraging. In particular, the authors were able to show that their greedy construction process for coding matrices outperforms random construction processes and reaches a compromise between sparsity and row/column dissimilarity.

4 Results

Table 2 presents the best results obtained for each task and each evaluation measure. One can observe here that the results in task 2, certainly the most difficult task as it involves non i.i.d data, are the lowest. One can also observe that the highest results are achieved in task 4 where all the information is provided to the systems. The best accuracy obtained in task 4 amounts to 50%, which is impressive given the complexity of the task. Finally, one can see that the results in task 1 and 3 are very close, a possible indication that description vectors bring little additional information over content vectors.

Figure 1 presents the macro-F_1 scores of all methods for task 1 and task 2. An arrow linking two or more result bars in figure 1 indicates that the difference between the corresponding systems is not statistically significant according to the S-test. For example, in task 1, the difference between the first two systems is not significant, whereas the difference between the 2^{nd} and 3^{rd} is. Note that for readability reasons, we did not compare all pairs of systems, but only the pairs corresponding to consecutive ranks for a given measure. As one can note, systems perform differently on tasks 1 and 2.

It is very interesting to note also that the two main approaches (*big-bang* and *midly hierarchical*) have provided state-of-the-art results on task 1 (a similar situation occurs in task 4). The ranks obtained by the different systems vary from one evaluation measure to the other. However, we have noticed a certain correlation between the results provided by the accuracy and the tree-induced error, in particular at the top of the result lists. Not surprisingly, the situation is different for the macro-F_1, as this measure focuses on a different view of the data in which categories are playing an equal role.

	Task 1	Task 2	Task 3	Task 4
Accuracy	0.467632	0.380619	0.467861	0.504759
Macro F-measure	0.35494	0.317133	0.359557	0.386195
Tree Induced Error	3.07858	3.80803	3.2621	2.82101

Table 2: Best results obtained on each task, for the three evaluation measures

The participation in the scalability tests was limited to four systems. This test has verified the difference between lazy and eager learners. For example, the *Turing* system, based on kNN, takes practically no time during training (less than a minute), but demands a lot of time for classification (around 21 hours). On the other hand, the *logicators* system, which uses SVMs, balances the time it demands between training (67 minutes) and classification (5 hours).

5 Conclusions

In general, we believe that the challenge and the workshop gave the opportunity to the research community to address and discuss the problem of large scale hierarchical classification. Furthermore, by providing the datasets and the oracle, both maintained at the LHSTC

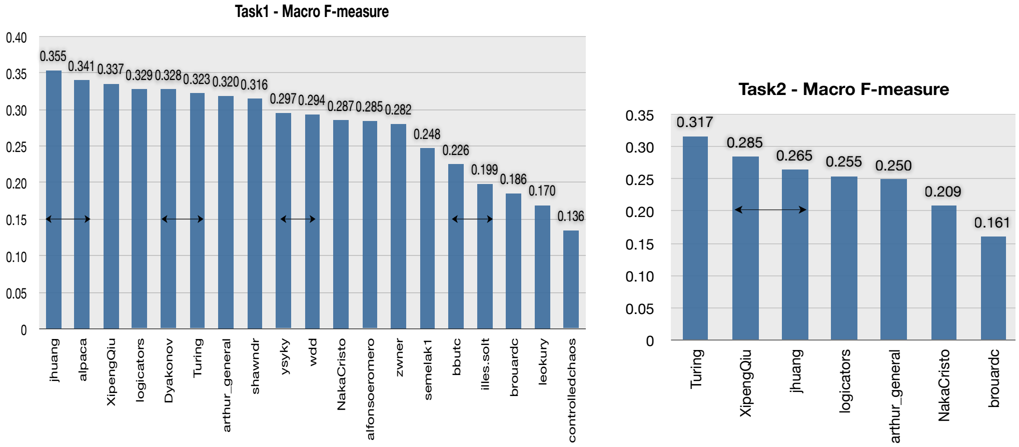

Figure 1: **Macro-F_1 results in task 1 and task 2**

website[2], the environment for further experiments is readily available to researchers. In the future, we are planing a second version of the challenge, that will have an increased number of categories and will incorporate new aspects, such as the use of multi-label data.

Acknowledgements This research was partly supported by the Pascal-2 Network of Excellence ICT-216886-NOE.

References

[1] A. Addis, G. Armano, and E. Vargiu. From a generic multiagent architecture to multiagent information retrieval systemst. In *AT2AI-6, Sixth International Workshop, From Agent Theory to Agent Implementation,*, pages 3–9, 2008.

[2] A. Addis, G. Armano, and E. Vargiu. Using the progressive filtering approach to deal with input imbalance in large-scale taxonomies. In *Large-Scale Hierarchical Classification Workshop of ECIR 2010*, 2010.

[3] J. Brank, D. Mladenic, and M. Grobelnik. Large-scale hierarchical text classification using svm and coding matrices. In *Large-Scale Hierarchical Classification Workshop of ECIR*, 2010.

[4] C. Brouard. Classification by resonance in an associative network. In *Large-Scale Hierarchical Classification Workshop of ECIR*, 2010.

[5] C. C. Coterillo. An adaptation of soucy and mineau weightin for large scale text classification. Website, 2010. `http://lshtc.iit.demokritos.gr/system/files/Nakacristo.pdf`.

[6] K. Crammer, O. Dekel, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:2006, 2003.

[7] O. Dekel, J. Keshet, and Y. Singer. Large margin hierarchical classification. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 27, New York, NY, USA, 2004. ACM.

[2] http://lshtc.iit.demokritos.gr/

[8] E. Gaussier and F. Pacull. Distributed categorizer for large category systems. In *Mining Massive Datasets for Security - Selected Proceedings of the MMDSS 2007 NATO workshop*. IOS Press, 2008.

[9] T.-Y. Liu, Y. Yang, H. Wan, H.-J. Zeng, Z. Chen, and W.-Y. Ma. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explorations*, 7(1):36–43, 2005.

[10] T.-Y. Liu, Y. Yang, H. Wan, Q. Zhou, B. Gao, H.-J. Zeng, Z. Chen, and W.-Y. Ma. An experimental study on large-scale web categorization. In A. Ellis and T. Hagino, editors, *WWW (Special interest tracks and posters)*, pages 1106–1107. ACM, 2005.

[11] O. Madani and J. H. 0002. On updates that constrain the features' connections during learning. In Y. Li, B. Liu, and S. Sarawagi, editors, *Proceedings of KDD*, pages 515–523. ACM, 2008.

[12] O. Madani, W. Greiner, D. Kempe, and M. R. Salavatipour. Recall systems: Efficient learning and use of category indices. In *In International Conference on Artificial Intelligence and Statistics (AISTATS*, 2007.

[13] O. Madani and J. Huang. Large-scale many-class prediction via flat techniques. In *Large-Scale Hierarchical Classification Workshop of ECIR*, 2010.

[14] H. Malik. Improving hierarchical svms by hierarchy flattening and lazy classification. In *Large-Scale Hierarchical Classification Workshop of ECIR*, 2010.

[15] Y. Miao and X. Qiu. Hierarchical centroid-based classifier for large scale text classification. Website, 2010. `http://lshtc.iit.demokritos.gr/system/files/XipengQiu.pdf`.

[16] P. Soucy and G. W. Mineau. Beyond tfidf weighting for text categorization in the vector space model. In L. P. Kaelbling and A. Saffiotti, editors, *IJCAI*, pages 1130–1135. Professional Book Center, 2005.

[17] D. Wang, W. Zhang, D. Xue, and Y. Yu. Deep classifier for large scale hierarchical text classification. Website, 2010. `http://lshtc.iit.demokritos.gr/system/files/Turing.pdf`.

[18] X.-L. Wang and B.-L. Lu. Improved hierarchical svms for large-scale hierarchical text classification challenge. Website, 2010. `http://lshtc.iit.demokritos.gr/system/files/LSHTC_wang-rev.pdf`.

[19] G.-R. Xue, D. Xing, Q. Yang, and Y. Yu. Deep classification in large-scale text hierarchies. In S.-H. Myaeng, D. W. Oard, F. Sebastiani, T.-S. Chua, and M.-K. Leong, editors, *SIGIR*, pages 619–626. ACM, 2008.

[20] B. K. Y. Yang, J. Zhang. A scalability analysis of classifiers in text. In *ACM SIGIR Conference*. ACM, 2003.

[21] Y. Yang and X. Liu. A re-examination of text categorization methods. pages 42–49. ACM Press, 1999.

[22] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, pages 412–420, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.