M. Tech. (Computer Science) Dissertation

Large Scale hierarchical text classification

A dissertation submitted in partial fulfillment
of the requirements for the award of
M.Tech.(Computer Science) degree

By

Gourab Saha

Roll No: MTC1109

under the supervision of

Professor Swapan Kumar Parui

Computer Vision and pattern recognition Unit

I N D I A N S T A T I S T I C A L I N S T I T U T E

203, Barrackpore Trunk Road

Kolkata - 700 108

Acknowledgements

At the end of this course, it is my pleasure to thank everyone who has helped me along the way.

First of all, I want to express my sincere gratitude to my supervisor Prof. Swapan Kumar Parui for his patience and advice and for the way he helped me think about problems with a broader perspective, I will always be grateful.

I would like to thank all the professors at ISI who have made my educational life exciting and helped me gain a better outlook on computer science. .

I would like to thank everybody at ISI for providing a wonderful atmosphere for pursuing my studies. I thank all my classmates who have made the academic and non-academic experience very delightful. It has been great having them around at all times, good or bad.

My most important acknowledgement goes to my family and friends who have filled my life with happiness. Most significantly to my parents who have always encouraged me to pursue my passions and instilled a love of knowledge in me; I am indebted all of them for their endless supply of encouragement, moral support and entertainment..

Abstract

Due to the growing amount of textual data, automatic methods for organizing the data are needed. Automatic text classication is one of this methods. It automatically assigns documents to a set of classes based on the textual content of the document.

Large-scale multi-labeled text classification is an emerging field because real web data have about several millions of samples and about half a million of non-exclusive categories. But this is a challenging task in that it is hard for a single algorithm to achieve both performance and scalability at the same time.

Normally, the set of classes is hierarchically structured but most of todays classication approaches ignore hierarchical structures, thereby loosing valuable human knowledge.This thesis exploits the hierarchical organization of classes to improve accuracy and reduce computational complexity.

Experiments are performed on Track 1 medium size wikipidia data set from ECML/PKDD 2012 discovery challenge. A top-down hierarchical classification method has been proposed using local classifier at each intermediate node.

Contents

To my Dear Parents.....

Chapter 1

Introduction

This chapter introduces the need for text classification in todays world and gives some examples of application areas. Problems of flat text classification compared to hierarchical text classification and how they may be solved by incorporating hierarchical information are outlined.

One common problem in the information age is the vast amount of mostly unorganized information. Internet and corporate Intranets continue to increase and organization of information becomes an important task for assisting users or employees in storing and retrieving information. Tasks such as sorting emails or files into folder hierarchies, topic identication to support topic-specific processing operations, structured search and/or browsing have to be fulfilled by employees in their daily work. Also, available information on the Internet has to be categorized somehow. Web directories like for example Yahoo are build up by trained professionals who have to categorize new web sites into a given structure.

Mostly this tasks are time consuming and sometimes frustrating processes if done manually. Categorizing new items manually has some drawbacks:

- 1. For special areas of interest, specialists knowing the area are needed for assigning new items (e.g. medical databases, juristic databases) to predefined categories.
- 2. Manually assigning new items is an error-prone task because the decision is based on the knowledge and motivation of an employee.

- 3.Decisions of two human experts may disagree (inter-indexing inconsistency).

Therefore tools capable of automatically classifying documents into categories would be valuable for daily work and helpful for dealing with todays information volume.

1.1 Application areas

To give a motivation for text classication, this section concludes with application areas for automatic text classication.

1.1.1 Automatic Indexing

Automatic Indexing deals with the task of describing the content of a document through assigning key words and/or key phrases. The key words and key phrases belong to a finite set of words called controlled vocabulary. Thus, automatic indexing can be viewed as a text classication task if each keyword is treated as separate class. Furthermore, if this vocabulary is a thematic hierarchical thesaurus this task can be viewed as hierarchical text classification.

1.1.2 Document Organization

Document organization uses text classification techniques to assign documents to a predefined structure of classes. Assigning patents into categories or automatically assigning newspaper articles to predefined schemes like the IPTC Code (International Press and Telecommunication Code) are examples for document organization.

1.1.3 Text Filtering

Document organization and indexing deal with the problem of sorting documents into predefined classes or structures. In text filtering there exist only two disjoint classes, relevant and irrelevant. Irrelevant documents are dropped and relevant documents are

delivered to a specic destination. E-mail filters dropping junk mails and delivering serious mails are examples for text filtering systems.

1.1.4 Word Sense Disambiguation

Word Sense Disambiguation tries to find the sense for an ambiguous word within a document by observing the context of this word (e.g. bank=river bank, nancial bank). WSD plays an important role in machine translation and can be used to improve document indexing.

Chapter 2

Problem Statement

The following section introduces definitions used in this thesis. For easier reading this section precedes the problem formulation.

2.1 Definations

Since the implemented algorithms are used to learn hierarchies some preliminary definitions describing properties of such hierarchies and their relationship to textual documents and classes are given.

2.1.1 Hierarchy

A Hierarchy $\mathcal{H} = (\mathbf{N}, \mathbf{E})$ is defined as directed acyclic graph consisting of a set of nodes $\mathbf{N}$ and a set of ordered pairs called edges $(\mathbf{N}_p, \mathbf{N}_c) \in \mathbf{N} \times \mathbf{N}$. The direction of an edge $(\mathbf{N}_p, \mathbf{N}_c)$ is defined from the parent node $\mathbf{N}_p$ to the direct child node $\mathbf{N}_c$. Additionally there exists exactly one node called root node $\mathbf{N}_r$ of a graph $\mathcal{H}$ which has no parent. Nodes which are no child nodes are called leaf nodes.Set of leaf nodes are called $\mathbf{N}_{leaves}$. All nodes except leaf nodes and the root node are called inner nodes.

2.1.2 Classes

Each node $\mathbf{N}_i$ within a hierarchy is assigned exactly to one class $\mathbf{C}_i(\mathbf{C} \equiv \mathbf{N} \in \mathcal{H})$. Set of leaf classes are called $\mathbf{C}_{leaves}$. Each leaf class consists of a set of documents $\mathbf{D}_i \in \mathbf{C}_i(\mathbf{C}_i \in \mathbf{C}_{leaves})$.

2.1.3 Documents

Documents of a hierarchy $\mathcal{H}$ contain the textual content and are assigned to one or more leaf classes. Classes of a document are also called labels of a document $\mathbf{L} =< \mathbf{C}_1, \mathbf{C}_2 \ldots \mathbf{C}_l >$.

In general each document is represented as term vector $\vec{d_i} =< d_{1,i}, d_{2,i} \ldots d_{n,i} >$ where each dimension $d_{j,i}$ represents the weight of a term obtained from preprocessing. Preprocessing methods are discussed in subsequent Sections.

2.2 Problem formulation

Since hierarchical text classification is an extension of flat text classification, the problem formulation for flat text classification is given first. Afterwards the problem definition is extended by including hierarchical structures which gives the problem formulation for this thesis.

2.2.1 Text Classification

Text Classifcation is the task of finding an approximation for the unknown target function $\psi : \mathbf{D} \times \mathbf{C} \rightarrow \{\mathbf{T}, \mathbf{F}\}$ where $\mathbf{D}$ is a set of documents and $\mathbf{C}$ is a set of predefined classes. Value $\mathbf{T}$ of the target function $\psi : \mathbf{D} \times \mathbf{C} \rightarrow \{\mathbf{T}, \mathbf{F}\}$ is the decision to assign document $\mathbf{D}_j \in \mathbf{D}$ to classes $\mathbf{C}_i \in \mathbf{C}$ and value $\mathbf{F}$ for not.

The approximating function $\tilde{\psi} : \mathbf{D} \times \mathbf{C} \rightarrow \{\mathbf{T}, \mathbf{F}\}$ is called classifier and should coincide with ψ as much as possible.

For the application considered in this thesis the following assumptions for the above definition are made:

- The target function ψ is described by a document corpus . A corpus is defined through the set of classes $\mathbf{C}$, the set of documents $\mathbf{D}$ and the assignment of classes to documents $\mathbf{D}_j \in \mathbf{C}$.
- Documents $\mathbf{D}$ are represented by a textual content which describes the semantics of a document.
- Categories $\mathbf{C}$ are symbolic labels for documents providing no additional information like for example meta data.
- Documents $\mathbf{D}_j \in \mathbf{D}$ can be assigned to more than one category (multi-label text classification). This is a special case of binary text classication.

For classifying documents automatically, the approximation $\tilde{\psi}$ has to be constructed.

2.2.2 Hierarchical Text Classification

Supplementary to the denition of flat text classification a graph $\mathcal{H}$ is added . $\mathcal{H}$ is a hierarchical structure defining relationships among classes.

The assumption is, that $\mathbf{C}_i \rightarrow \mathbf{C}_j$ defines a IS-A relationship among classes whereby $\mathbf{C}_i$ has a broader topic than $\mathbf{C}_j$ and the topic of a parent class covers all topics from all of its child classes($\forall \mathbf{C}_k, \mathbf{C}_i \rightarrow \mathbf{C}_k$).The IS-A relationship is asymmetric (e.g. all dogs are animals, but not all animals are dogs) and transitive (e.g. all pines are evergreens and all evergreens are trees; therefore all pines are trees). The goal is, as before, to approximate the unknown target function by using a document corpus.

Since classification methods depend on the given hierarchical structure including classes and assigned documents, the following basic properties can be distinguished:

- Structure of the hierarchy:

Given the above general denition of a hierarchy $\mathcal{H}$, two basic cases can be distinguished. (i) A tree structure, where each class (except the root class) has exactly one parent class and (ii) a directed acyclic graph structure where a class can have more than one parent classes.

- Classes containing documents:

  Another basic property is the level at which documents are assigned to classes within a hierarchy. Again two different cases can be distinguished. In the first case, documents are assigned only to leaf classes . In the second case a hierarchy may also have documents assigned to inner nodes. Note that the later case can be extended to the previous one by adding a virtual leaf node to each inner node. This virtual leaf node contains all documents of the inner node.

- Assignment of documents

  As done in at text classification, it can be distinguished between multi label and single label assignment of documents. Depending on the document assignment the classification approach may differ.

The model proposed here is a top-down approach to hierarchical text classification by using a directed acyclic graph. Additionally, multi label documents are allowed. A top down approach means that recursively, starting at the root node, at each inner node zero, one or more subtrees are selected by a local classifier. Documents are propagated into these subtrees till the correct class(es) is/are found.

Chapter 3

A top-down algorithm for hierarchical text classification

3.1 Overview

In this thesis, I propose a top-down hierarchical classification apporoach. My apporoach is based on similarity between test document and each inner class. Each test will be given to root node and at each inner node it's classified using local classifiers.If similarity between test document and an inner node is found less, subtree rooted at that particular inner node is not explored further . This process continues until the test document propagates to one or more leaf nodes.

3.2 Hierarchy Representation

A hierarchy $\mathcal{H}$ of categories is a collection of superior categories (superiors or parents), each of which subsumes a collection of subordinate categories (subordinates or children).Each subordinate could have its own subordinates, until the most specific categories (leaf categories) are reached. Legal categories for the classification task are only those leaf categories in the hierarchy which do not have any subordinates.

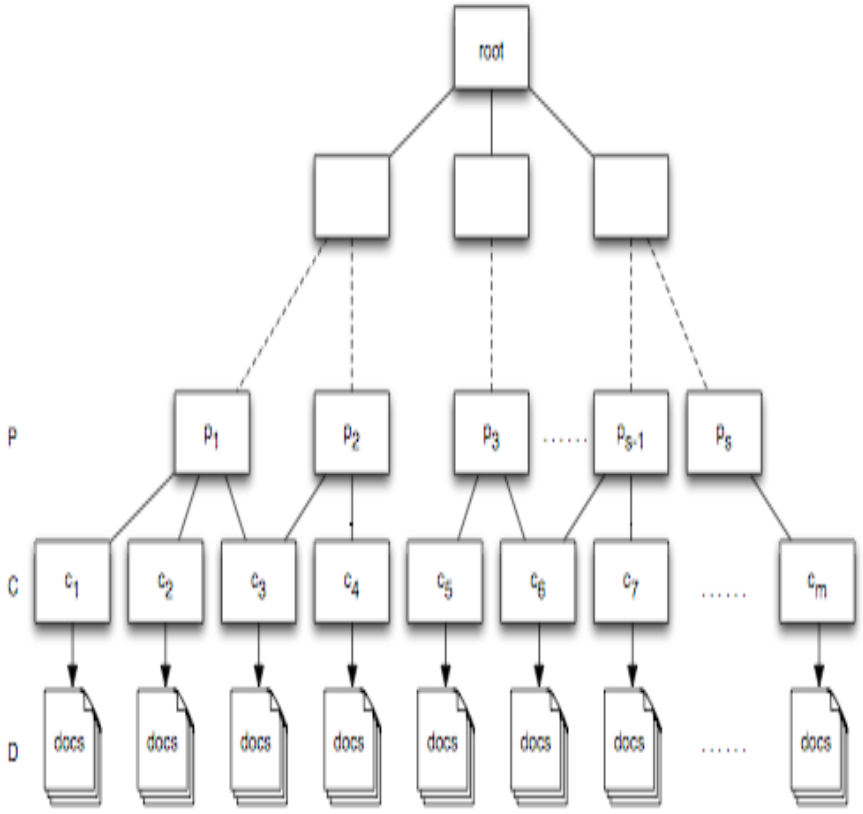

FIGURE 3.1: *Hierarchical structure*

Our assumption of the category hierarchy is, if a document belongs in a category c, it also belongs in each of parent categories of c

for a given node n_i parents(n_i)=Set of all parents of node n_i and children(n_i)=Set of all children of node n_i.

3.3 Document Representation

Document representation is the step of mapping the textual content of a document into a logical view which can be processed by classification algorithms. The logical view of a document d_j can be obtained by extracting all meaningful units(terms) from all documents and assigning weights to each term in a document reflecting the importance of a term within the document. More formally, each document is assigned a n-dimensional

vector $\vec{d_j} =< (t_1, w_1), (t_2, w_2) \ldots (t_n, w_n) >$ using vector space model where each t_i is a term from Term set $\mathcal{T}$ AND w_i is it's corresponding weight. Obtaining the vector representation involves two major steps.

3.3.1 Preprocessing

Stopwords, which are topic neutral words such as articles or prepositions contain no valuable or critical information. These words can be safely removed, if the language of a document is known. Removing stopwords reduces the dimensionality of term space. On the other hand a sophisticated usage of stopwords (e.g. negation, prepositions) can increase classification performance.

One problem in considering single words as terms is different syntactical forms may describe the same word (e.g. go, went, walk, walking). Stemming is the notation for reducing words to their root form.For English a lot of stripping and stemming algorithms exist, the Porters Algorithm being the most popular one.

In the preprocessing phase we have dropped all the stop words and converted all terms to its root form using Porters Stemmer algorithm.

3.3.2 Term weighting

Initially for each catagory $\mathbf{C}_i \in \mathbf{C}_{leaves}$ we have a set of vector

$D_i = \{\vec{d_1}, \vec{d_2} \ldots \vec{d_k}\}$.

we define a single vector for each $\mathbf{C}_i \in \mathbf{C}_{leaves}$,

$$\vec{D_i} = \sum_{\forall \vec{d_k} \in D_i} \vec{d_k} \tag{3.1}$$

where weight of term t_j in class $\mathbf{C}_i$

$$w_{j,\vec{D_i}} = \sum_{\forall \vec{d_k} \in D_i} w_{j,k} \tag{3.2}$$

After extracting the term space from a document corpus the influence of each term within a document has to be determined. Therefore each term within a document is assigned a weight leading to the above described vector representation.

$$\vec{D_j} =< (t_1, w_{j,1}), (t_2, w_{j,2}) \ldots (t_n, w_{j,n}) > \tag{3.3}$$

where t_i reresents the term and $w_{j,i}$ represents its corresponding weight. Initially $w_{j,i}$=frequency of term t_i in $\vec{D_j}$ .For every document $\vec{D_j}$ It has been further normalized independently using

$$w_{j,i} = w_{j,i} / \arg\max(w_{j,i}) \tag{3.4}$$

3.4 Bottom up propagation

Since we have training data only at the leaf nodes of the hiereachy we have to propagate it from bottom to top. For each inner and root classes $\mathbf{C}_i$ we define a single vector

$$\vec{D_i} = \sum_{\forall \mathbf{C}_k \in children(\mathbf{C}_i)} \vec{C_k} \tag{3.5}$$

where weight of term t_j in class $\mathbf{C}_i$ is

$$w_{j,i} = \sum_{\forall \mathbf{C}_k \in children(\mathbf{C}_i)} w_{j,k} \tag{3.6}$$

Starting from the leaf node it gradually go upto the root. In figure 3.2 initially leaf classes D,E,F have $\{X1, X2\}, \{X1\}, \{X3\}$ set of documents respectively. The document set at B is created taking the union of document set of its children D and E. Similarly document set at C is created and which are further propagated to root A.

3.5 Dimensionality reduction

The approach on dimensionality reduction by term selection is the called filtering approach. Thereby measurements derived from information or statistical theory are used

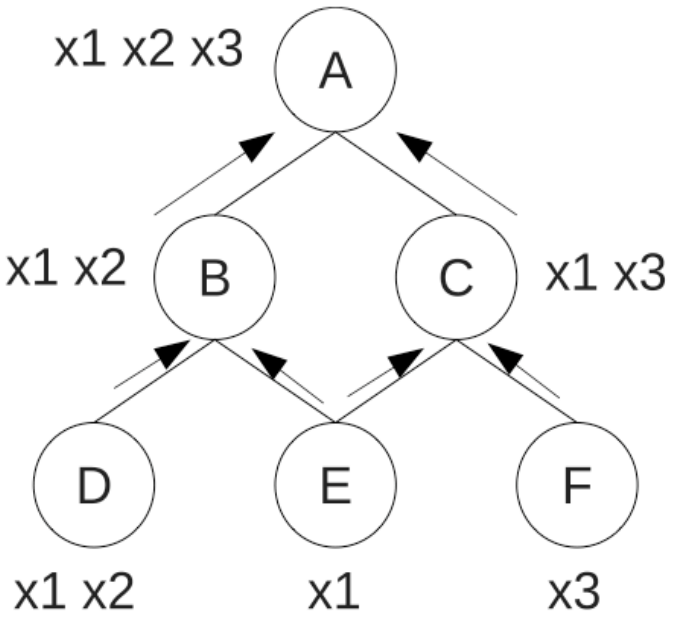

FIGURE 3.2: *Bottom up propagation of training data*

to filter irrelevant terms. Afterwards the classifier is trained on the reduced term space. In this approach dimensionality reduction of vectors for each node is done indipentently.

For each inner and root class $\mathbf{C}_i$ for each term t we calculate $idf(t, \mathbf{C}_i)$ is defined as

$$idf(t, \mathbf{C}_i) = \log_{10}(\frac{n - n_t + 0.5}{n_t + 0.5}) \tag{3.7}$$

where $n = |children(\mathbf{C}_i)|$

$n_t = |\{\mathbf{C}_j | w_{t,\vec{D_j}} \neq 0, \mathbf{C}_j \in children(\mathbf{C}_i)\}|$

For each class $\mathbf{C}_i$ other than root for each term $t_k$we calculate $tfidf(t_k, \mathbf{C}_i)$ defined as

$$tfidf(t_k, \mathbf{C}_i) = w_{t,\vec{D_i}} * \arg\max\{idf(t_k, \mathbf{C}_j), \forall \mathbf{C}_j \in parents(\mathbf{C}_i)\} \tag{3.8}$$

A particular term t_k is dropped from class $\mathbf{C}_i$ if

$$tfidf(t_k, \mathbf{C}_i) / \arg\max\{tfidf(t_j, \mathbf{C}_i), \forall t_j \in \mathbf{C}_i\} < \theta \tag{3.9}$$

$\theta \in [0, 1]$ and chosen experimentally . In this approach $\theta = 0.2$ has been used.

3.6 Similarity measure

In this thesis I used introduce The BM25 as a measures to calculate the similarity between a test document and a class.

$$bm25sim(\vec{d_j}, \mathbf{C}_i) = \sum_{\forall t_k \in \vec{d_j}} \arg\max\{idf(t_k, \mathbf{C}_j), \forall \mathbf{C}_j \in parents(\mathbf{C}_i)\} \frac{w_{t_k, \mathbf{C}_i} * (k_1 + 1)}{w_{t_k, \mathbf{C}_i} + k_1 * (1 - b + b * \frac{length(D_i)}{avg_{length}})} \tag{3.10}$$

where avg_{length} =Average no of terms per class.
$length(D_i)$ =No of terms in class $\mathbf{C}_i$

In this experiment, k1 = 1.5 and b = 0.75 has been taken. $idf(t_k, \mathbf{C}_i)$ is the inverse document frequency of the term t_k in $\mathbf{C}_i$ and computed as:

$$idf(t_k, \mathbf{C}_i) = \log_{10}(\frac{n - n_t + 0.5}{n_t + 0,5}) \tag{3.11}$$

3.7 Multiple Category Classification

As each document can be assigned to multiple categories in the hierarchy, we select top-M categories as the predicted categories of a query document d. Note M varies across documents, so one problem is how to decide M for each document. Let *avglabels* denote the average number of leaf categories per document within the hierarchy, which is pre-computed from the training set. For the ranked list of categories $(rs(c_{i1}), rs(c_{i2}), ...rs(c_{ik}), ..)$ computed by the algorithm at each intermidiate node, we

choose all categories whose ranking scores are large enough relative to the largest score $rs(c_{i1})$ i.e

$$rs(c_{ik})/rs(c_{i1}) > \alpha \tag{3.12}$$

where $0 \leq \alpha \leq 1$

In order to tune α, we calculate the predicted average number of categories per document in the test set deonted as $avgPredlabels(\alpha)$ By iteratively trying dierent values of α, and calculating the $error = |avgPredlabels(\alpha) - avglabels|$, the α value with the minimum error is chosen as the ratio threshold.

3.8 Algorithm

Algorithm 1 Classification of Test document

Procedure Classify(d_j)

```
1: CREATEQUE(Q)
2: ENQUE(Q,root)
3: while NOTEMPTY(Q) do
4:     Node=DEQUE(Q)
5:     Mark Node VISITED
6:     for All Children c_i of Node do
7:         Find Ranked list based on similarity score with d_j(rs(c_i1), rs(c_i2), ...rs(c_ik), ..)
8:         if (rs'(c_ik)/rs'(c_i1)) > α then
9:             if c_ik is a leaf node then
10:                OUTPUT c_ik
11:            else
12:                if c_ik isNOT VISITED then
13:                    ENQUE(Q,c_ik)
14:                end if
15:            end if
16:        end if
17:    end for
18: end while
```

EndProcedure

Chapter 4

Experiment and Result

4.1 Dataset

In ECML/PKDD 2012 Discovery Challenge track 1 consists a large dataset created from Wikipedia. The datasets are multi-class, multi-label and hierarchical.

Dataset contains trainset ie- documents with labels, hierarchy information and testset ie- documents without labels.

- No of leaf levelcatagories: 36504
- No of total catagories : 50312
- No of Train documents :456886
- No of Test documents :81262

Indegree and outdegree distributions of the given hierarchies is given in the figure.

4.2 Metrics for eavaluation

Various performance measures within text classification exist, covering different aspects of the task. This section covers the most used performance measures, their benifits

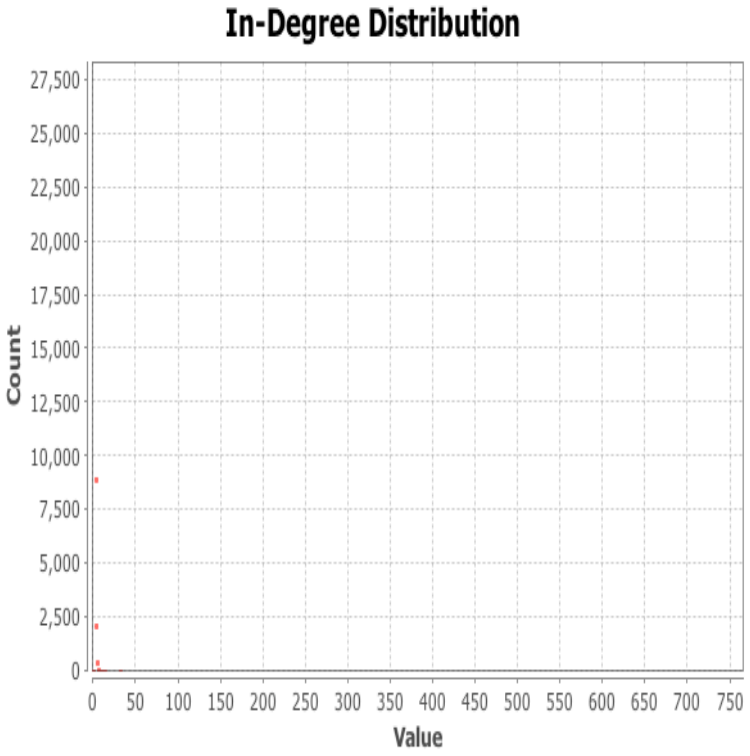

FIGURE 4.1: *Indegree distribution data*

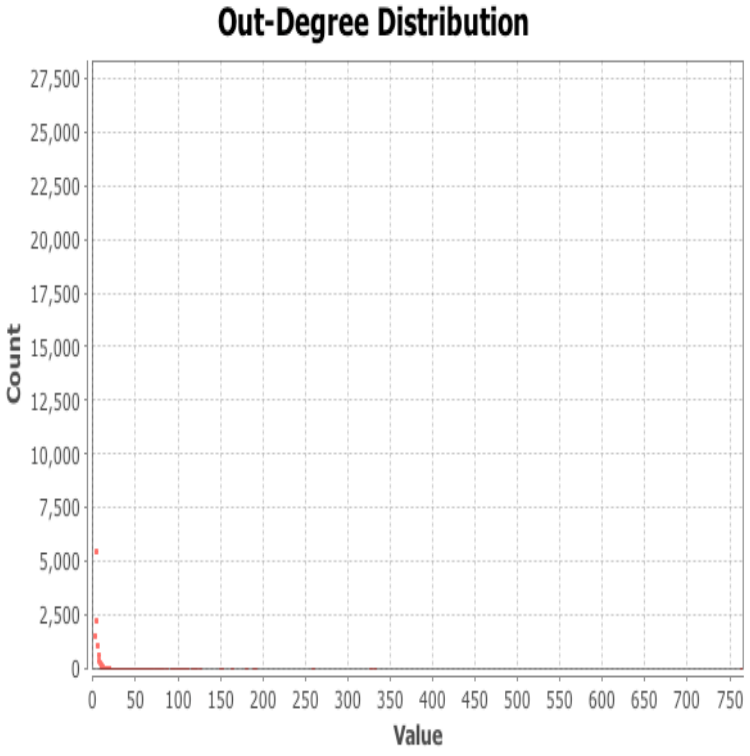

FIGURE 4.2: *Outdegree distribution*

and drawbacks.The Craneld tests, conducted in 1960s, established the desired set of characteristics for a retrieval system. Even though there has been some debate over the years, the two desired properties that have been accepted by the research community for measurement of search effectiveness are recall,i.e., the proportion of relevant documents retrieved by the system; and precision, i.e., the proportion of retrieved documents that are relevant.

4.2.1 Precision and recall

Eectiveness is purely a measure of the ability of the system to satisfy the user in terms of the relevance of documents retrieved. Initially, eectiveness can be measured exploiting precision and recall; a similar analysis could be given for any pair of equivalent measures. It is helpful at this point to introduce the famous confusion matrix (also called contingency table in the IR context) depicted in table.

TABLE 4.1: Precision and Reacll

Documents	Deemed non-relevent	Deemed relevent
negative	true negative (TN)	false positive (FP)
positive	false negative (FN)	true positive (TP)

Such table is a visualization tool typically used in supervised learning (where it is also called a matching matrix). Each row of the matrix represents the instances in a predicted class, while each column represents the instances in an actual class. One benet of a confusion matrix is that it is easy to see if the system is confusing two classes (i.e., commonly mislabeling one as another). In an information retrieval scenario, Precision is defined as the number of relevant documents retrieved by a search divided by the total number of documents retrieved by that search (namely precision = T P /(T P + F P)), and Recall is defined as the number of relevant documents retrieved by a search divided by the total number of existing relevant documents (which should have been retrieved, namely reacall = T P /(T P + F N)). It is well accepted that a good IR system should retrieve as many relevant documents as possible (i.e., have a high recall), and it should retrieve very few non-relevant documents (i.e., have high precision). Unfortunately, these two goals have proved to be quite contradictory over the years. Techniques that

tend to improve recall tend to hurt precision and vice-versa; for example, if system designers feel that precision is more important to their users, they can use precision in top ten or twenty documents as the evaluation metric. On the other hand, if recall is more important to users, one could measure precision at (say) 50 % recall, which would indicate how many non-relevant documents a user would have to read in order to nd half the relevant ones.

4.2.2 Combining Precision and recall

There are techniques allowing to combine all these values in order to find an evaluation that wraps all the information about how well a system is performing.

An obvious method that may occur to the reader is to judge an information retrieval system by its accuracy, that is, the fraction of its classications that are correct. In terms of the confusion matrix above, accuracy = (T P + T N)/(T P + F P + F N + T N). This seems plausible, since there are two actual classes, relevant and non-relevant, and an information retrieval system can be thought of as a two-class classifier which attempts to label them as such (it retrieves the subset of documents which it believes to be relevant). This is precisely the eectiveness measure often used for evaluating machine learning classication problems. There is a good reason why accuracy is not an appropriate measure for information retrieval problems. In almost all circumstances, the data is extremely skewed: normally over 99.9relevant to all queries. Even if the system is quite good, trying to label some documents as relevant will almost always lead to a high rate of false positives. However, labeling all documents as non-relevant is completely unsatisfying to an information retrieval system user. Users are always going to want to see some documents, and can be assumed to have a certain tolerance for seeing some false positives providing that they get some useful information. The measures of precision and recall concentrate the evaluation on the return of true positives, asking what percentage of the relevant documents have been found and how many false positives have also been returned. The advantage of having the two numbers for precision and recall is that one is more important than the other in many circumstances. Typically, web surfers would like every result on the rst page to be relevant (high precision) but

have not the slightest interest in knowing let alone looking at every document that is relevant. In contrast, various professional searchers such as paralegals and intelligence analysts are very concerned with trying to get as high recall as possible, and will tolerate fairly low precision results in order to get it. Individuals searching their hard disks are also often interested in high recall searches. Nevertheless, the two quantities clearly trade o against one another: you can always get a recall of 1 (but very low precision) by retrieving all documents for all queries! Recall is a non-decreasing function of the number of documents retrieved. On the other hand, in a good system, precision usually decreases as the number of documents retrieved increase. In general, we want to get some amount of recall while tolerating only a certain percentage of false positives.

4.2.3 F-measure

A single measure that trades o precision versus recall is the F-measure based on a on van Rijsbergens eectiveness measure. The F measure is the weighted harmonic mean of precision and recall:

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(1+\beta^2)(precision * recall)}{(\beta^2 * precision * recall)} where \beta^2 = \frac{1-\alpha}{\alpha} \tag{4.1}$$

The default balanced F measure equally weights precision and recall, which means making $\alpha = 1/2$ or $\beta = 1$. When using $\beta = 1$, the formula on the right simplies to:

$$F = \frac{2PR}{P+R} \tag{4.2}$$

The metrics used for evaluating the classication algorithms include accuracy,precison ,recall, example-based F-measure, label-based macro F-measure, label-based micro F-measure

4.3 Result

The results of our algorithm for the multi-task learning track (Track 1) are shown in Table . It shows that our algorithm produced high accuracy f compared with k-NN baseline. However, the performance for the Wikipedia data is relatively lower which might due to the noise in the Wikipedia data set.

TABLE 4.2: Result

Name	Acc	EBF	EBP	EBR	LBMaF	LBMaP	LBMaR
Best	0.438162	0.493725	0.551626	0.496298	0.267413	0.573306	0.287564
MyResult	0.407741	0.446041	0.50381	0.432612	0.238535	0.489009	0.244664
KnnBaseline	0.249137	0.317596	0.282953	0.41639	0.175792	0.252206	0.235399

- Acc : Accuracy
- EBF :F1-measure
- EBP :Precision
- EBR :Recall
- LBMaF: Label based Macro F1-measure
- LBMaP:Label based Macro Precision
- LBMaR:Lbel based Macro Recall

Chapter 5

Conclusion

In this thesis I proposed a hierarchical text classication method based on BM-25.Firstly trainset documents are propagated from bottom to top upto the root. Secondly, important candidate category features were extracted. Finally, the categories prediction algorithm uses a top-down approach and use BM25 similarity measure to assign scores to the candidate categories, and the top ranked categories are chosen as the predicted categories of the query document. Different costant parameter has been chosen experimentally.

Chapter 6

Bibliography

Hierarchical text classification (inproceedings) Author Pulijala, Ashwin and Gauch, Susan Booktitle International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA Year 2004 Pages 21–25 Organization Citeseer

Hierarchical text classification and evaluation (inproceedings) Author Sun, Aixin and Lim, Ee-Peng Booktitle Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on Year 2001 Pages 521–528 Organization IEEE

Improving Text Classification by Shrinkage in a Hierarchy of Classes. (inproceedings) Author McCallum, Andrew and Rosenfeld, Ronald and Mitchell, Tom M and Ng, Andrew Y Booktitle ICML Year 1998 Volume 98 Pages 359–367

Hierarchical classification of Web content (inproceedings) Author Dumais, Susan and Chen, Hao Booktitle Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval Year 2000 Pages 256–263 Organization ACM

Learning hierarchical multi-category text classification models (inproceedings) Author Rousu, Juho and Saunders, Craig and Szedmak, Sandor and Shawe-Taylor, John Booktitle Proceedings of the 22nd international conference on Machine learning Year 2005 Pages 744–751 Organization ACM

Feature selection for classification based on text hierarchy (inproceedings) Author Mladeni'c, Dunja and Grobelnik, Marko Booktitle Text and the Web, Conference on Automated Learning and Discovery CONALD-98 Year 1998 Organization Citeseer

gopalregularization Regularization Framework for Large Scale Hierarchical Classification (article) Author Gopal, Siddharth and Yang, Yiming and Niculescu-Mizil, Alexandru

The ECIR 2010 large scale hierarchical classification workshop (inproceedings) Author Kosmopoulos, Aris and Gaussier, Eric and Paliouras, Georgios and Aseervatham, Sujeevan Booktitle ACM SIGIR Forum Year 2010 Volume 44 Pages 23–32 Number 1 Organization ACM

Hierarchical Text Classification with Latent Concepts. (inproceedings) Author Qiu, Xipeng and Huang, Xuanjing and Liu, Zhao and Zhou, Jinlong Booktitle ACL (Short Papers) Year 2011 Pages 598–602

Enhanced K-Nearest Neighbour Algorithm for Large-scale Hierarchical Multi-label Classification (inproceedings) Author Wang, Xiao-lin and Zhao, Hai and Lu, Bao-liang Booktitle Proceedings of the Joint ECML/PKDD PASCAL Workshop on Large-Scale Hierarchical Classification, Athens, Greece Year 2011 Volume 5

A k-NN Method for Large Scale Hierarchical Text Classification at LSHTC3 (article) Author Han, Xiaogang and Li, Shaohua and Shen, Zhiqi

Multi-Stage Rocchio Classification for Large-scale Multi-labeled Text data (article) Author Lee, Dong-Hyun

Evaluation Measures for Hierarchical Classification: a unified view and novel approaches (article) Author Kosmopoulos, Aris and Partalas, Ioannis and Gaussier, Eric and Paliouras, Georgios and Androutsopoulos, Ion Journal arXiv preprint arXiv:1306.6802 Year 2013

Regularization Framework for Large Scale Hierarchical Classification (article) Author Gopal, Siddharth and Yang, Yiming and Niculescu-Mizil, Alexandru

Field-weighted XML retrieval based on BM25 (incollection) Author Lu, Wei and Robertson, Stephen and MacFarlane, Andrew Booktitle Advances in XML Information Retrieval and Evaluation Publisher Springer Year 2006 Pages 161–171