Hypertext Classification

Diploma thesis

Hervé Utard

Supervisor
Professor Johannes Fürnkranz

May 15, 2005

Abstract

Web Directories have been historically collected and updated by hand but this method is unsatisfactory for three reasons: A team of Web Surfers maintaining such a database should face the gigantism of the World network. Its size would thus be incompatible with the economic constraints of startups. Even the biggest team would not be able to trace all the changes on the Web and to keep the database up to date. Furthermore, categorization is a highly subjective task. Manual categorization is not a synonym for good categorization. However, automating the categorization of documents is a difficult task in the Web environment. The diversity of languages, topics and authorships prevents the traditional classification algorithms to work optimally. Fortunately, the internal HTML structure of the Web pages and the hyperlink graph structure of the Web are new sources of information that can be explored to improve automated Web page classification.

In this diploma thesis, we carry on the work of Fürnkranz [8] about hypertext categorization. We investigate different classification techniques for categorizing hypertext documents. We target information rich text areas of the page and of its neighbors and we compare different methods for having those various features optimally help together for improved classification.

We evaluate the heavy points and the weaknesses of the *Hyperlink Ensembles* and *Meta Predecessor* approaches. We explain how to choose a binarization algorithm between *Round Robin* and *One Against All* according to the behavior awaited. We compare two solutions for bringing features mined on different locations together, namely *Tagging* and *Merging* and we finally propose a model of hypertext classifier which combines the best characteristics of the methods we study. Our main result is a model of hyperlink based classifier that outperforms a text only classifier by almost 25% for the WebKB dataset.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The quality of a library is not only measured by its completeness but also by the accessibility of the information. A reader searching a book can follow the classification by themes or can easily find a book of a given author because they are sorted alphabetically. If the reader still cannot find the book he looks for, he can ask a librarian who has an overview on the books collection and who has the knowledge to help the reader better expressing his wishes.

The Internet can be seen as the largest library in the world. A web user can find a book or web site instantaneously if he knows its name or Uniform Resource Locator (*URL*), what corresponds to the alphabetical sorting of the books' authors. He can also use a Web Catalogue (a.k.a Web Directory) to iteratively narrow his search like a reader would do with the classification of a library. Moreover, Search Engines have been developed to play the role of the librarians. But there is no exhaustive list of the pages available on the Web. Even the most complete Web Catalogues only reference a little subset of the Web [11]. The Search Engines reference larger subsets of the Web but like Web Catalogues, they ignore significant domains of the Web. And despite attempts to provide support for query refinement whereby users receive suggestions about terms to include or exclude from their query, the Search Engines are still far from being the equivalent of human librarians.

1.1 Search engines and Web Directories

Even though the first few web sites appeared in 1993, the premises of the search engines are to be found earlier. Alan Emtage wrote in 1990 the first search engine [3, 15]. It combined a script gathering the names of the files available on public FTP servers and a regular expression matcher for retrieving filenames matching a user query. The University of Nevada developed in 1993 a similar search engine working on plain text files. Web crawler, released in April 1994 was the first search engine that indexed entire web pages. It gave birth to Excite, Lycos, Infoseek and Opentext. The most powerful search engine to date, Google, was launched in 1998. Its success is due to its PageRank algorithm that incorporates information of the hyperlink structure to evaluate the degree of relevance of each answer. The open source Nutch project was started in 2003 in order to counterbalance the commercial search engines. Nutch is to the date when this document was written under active development. Some challenges for the search engines are to answer the queries as quick as possible and to circumvent keywords ambiguities (polysemy).

Web Directories organize a small subset of the web material into a hierarchy of thematic categories. The web site www.directoryarchives.com claims to be the directory of the directories. The web user chooses successively more and more specific topics until he gets a list of pages corresponding to his request. The categorization of web pages is a difficult task because deciding whether a document should be classified under a given category requires an understanding of the meaning of both the document and the category. Automatic categorization of documents is an active research area, but the major web directories have been historically manually collected and maintained. This historical choice is now criticized for various reasons: The growth of the Web is too fast so that a reasonably big team can insert the new pages into the web directory and take the changes of the pages already referenced into account. The categorization of a document is highly subjective. Classifying a newspaper article about the US military operations in Irak under the category *Iraqi Freedom* or under the category *Occupation* depends on the point of view of the reader. There is thus no guarantee that a manual categorization is a good categorization.

1.2 Web Mining

There is an intense activity in the Web Mining community to improve the performance of the browsing assistance tools. This research area is basically data mining for data on the World-Wide Web. It combines text, structure and usage mining. Web Mining is oriented to the formation or update of Web Catalogues, to the ranking or clustering of search results, to information extraction with the development of a world wide knowledge base or even to click stream analysis and product recommendation.

One research domain of Web Mining is automatic document classification. Motivation for this research area is firstly to build automatically Web directories. Those web directories could be as complete and up to date as the search engines because the limitation induced by the manpower needs wouldn't exist anymore. Secondly, automated document classification could grant the search engines indexing algorithms that would prevent the problems induced by polysemy and that would reduce the response time by associating to each document referenced keywords representative of the content of the document.

1.3 Text Classification

Some researchers like Sebastiani [16] define Text Classification (a.k.a Text Categorization) as the task of predicting if a given document is related with a given category. This is called binary classification or concept learning. In our study, we manipulate documents that must be assigned to exactly one category. Thereby, we use the definition of single-label classification which is the task of predicting the category that is related with a document.

More formally, classification is the task of approximating the unknown target function $\check{\Phi}$ (that describes how documents ought to be classified) by means of a function Φ called the classifier (a.k.a hypothesis) such that $\check{\Phi}$ and Φ coincide as much as possible (figure 1.1).

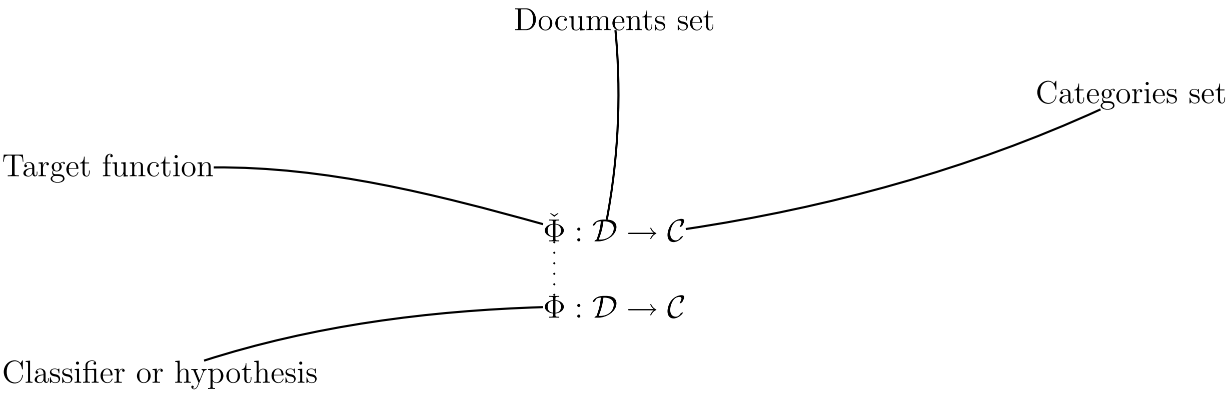

Figure 1.1: The classification problem

This subfield of the information systems discipline is born in the early '60s. Solving a Text Categorization problem was then costly because the most popular approach was to ask a human expert to define manually a set of rules encoding his knowledge (Figure 1.2).

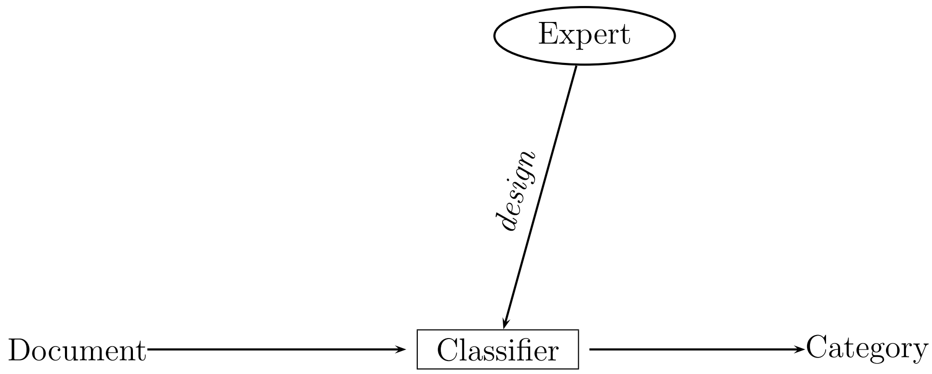

Figure 1.2: Expert designed classifier

In the late '80s, this approach was supplanted by the Machine Learning paradigm which consists of extracting inductive knowledge from the content of a set of documents pre-classified (Figure 1.3).

Since the early '90s, numerous models for inductive classifiers have been imagined. Probabilistic classifiers like the Bayes classifiers base their predictions on statistics computed from the frequencies of appearance of the words in the documents. The decision rule classifiers define hypothesis similar to those previously written by human experts while

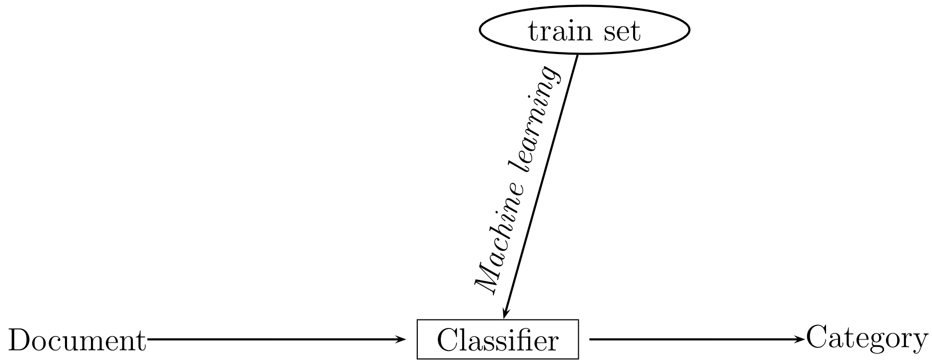

Figure 1.3: Machine learned classifier

decision trees organize them so that the successive rules refine the classification. While the attempts to clone human intelligence with the help of neural networks produced classifiers suffering from a bad accuracy, one of the most interesting models are linear Support Vector Machines. It distributes the examples in a multidimensional vector space and tries to find hyperplanes which separate optimally the different categories.

Obvious advantages of automated text classification are that no human expert manpower is needed to elaborate the classification hypothesis and that the models of classifiers are problem independent and can thus easily be adapted to any new classification problem. Moreover, the major quality of the machine learned classifiers is that they attain an accuracy comparable to that reached by the human experts designed classifiers.

Application fields for Text Categorization are spam filtering, automated indexing of digital libraries where the goal is to associate to each document the subset of keywords which describes its content from a predefined list of keywords, automated filing of newspaper articles under the appropriate sections or even word sense disambiguation for polysemous words like *just* or *stand*

Despite a quite good accuracy, the statistical models presented above are limited because they are based only on statistics computed from the word occurrences. There is currently an intense activity in the text classification and linguistic communities to develop models of Natural Language Processing classifiers which would get closer to the semantic of the documents, what should result in high classification performance levels.

1.4 Hypertext classification

The text classification methods described in the preceding section have unfortunately a poor accuracy when they are employed in the World Wide Web hypertext context. This is due to the heterogeneousness of the Web. Never had a database been fed by so many authors, in so many languages, about so many topics. Furthermore, the facilities brought by the Hypertext Markup Language (HTML) have resulted in a impoverishment of language. Many web page authors prefer for example use HTML lists instead of writing link words in an enumeration, which prevents natural language processing algorithms from correctly understanding the syntax and thus the semantic of the documents.

In a similar manner, the adage *One picture is worth a thousand words* is widely applied by web authors. This results in the existence of web pages containing no text but just a picture. Other numerous pages own an irrelevant content like *Page under construction* that can of course not be used for the classification.

Despite these difficulties, hypertext classification has big chances to outperform text classification: The loss of grammar structure brought on by the use of HTML is not a loss of structuring in the documents. The global structure of the documents has on the contrary increased with the facility to highlight an important word, to associate hierarchically structured headlines to the paragraphs, to gather word groups whose meaning or function is similar into a common list, to associate keywords to a page, to mention the author and the date of publication in fields especially designed for this purpose in the heading of the web documents.

But the most important progression with hypertext is the linkage between the documents. Not only the documents own an internal structure, but the whole database is organized in a graph structure. Hypertext Classification mines information not only in the content of the documents to classify but also in their hyperlink neighborhood.

Chapter 2

Related Work

In this chapter, we present the pionner work of Chakrabarti[5] about an enhanced hypertext categorization using hyperlinks and the work of Getoor and Lu [12] about Link Mining which is closely related to our study.

2.1 Enhanced hypertext categorization using hyperlinks

The first research on hypertext categorization using both local and linkage information has been led at IBM Almaden by Soumen Chakrabarti, Byron Dom and Piotr Indyk in 1998. They notice in [5] that the extreme diversity of the web documents prevents text classifiers to reach satisfactory performance levels while rich information can be mined in the broader context of the local region of hypertext documents. They first try to naively extend a traditional text classifier: They append the content of the neighbors at the end of the page to classify and use those meta-documents in the classification. This increases the error rate because the term distribution of the neighbors is not sufficiently similar to the distribution of the document class.

They further tag the terms mined in the neighborhood in order to distinguish them from the original local terms but it does not help because it splits terms into many forms making them relatively rare. The classifier is further challenged with many more features but not more documents. Even if some proposals could reduce the sources of noise, the authors prefer to explore a new way of hypertext categorization. They use class information from the neighbors to pilot an iterative relaxation labeling model (figure 2.1). For bootstrapping this relaxation labeling algorithm, only local terms are used to express a first categorization prediction. Once a class has been assigned to each member of the dataset, the relaxation phase starts with the hyperlink classifier which uses the class predictions of the neighbors of each document δ to correct its classification.

This second method gives interesting results. It reduces the error rate of a text-based classifier by over 70%. We believe however that going with the majority prediction of the neighbors can in some cases lead to bad conclusions. The home page of a university

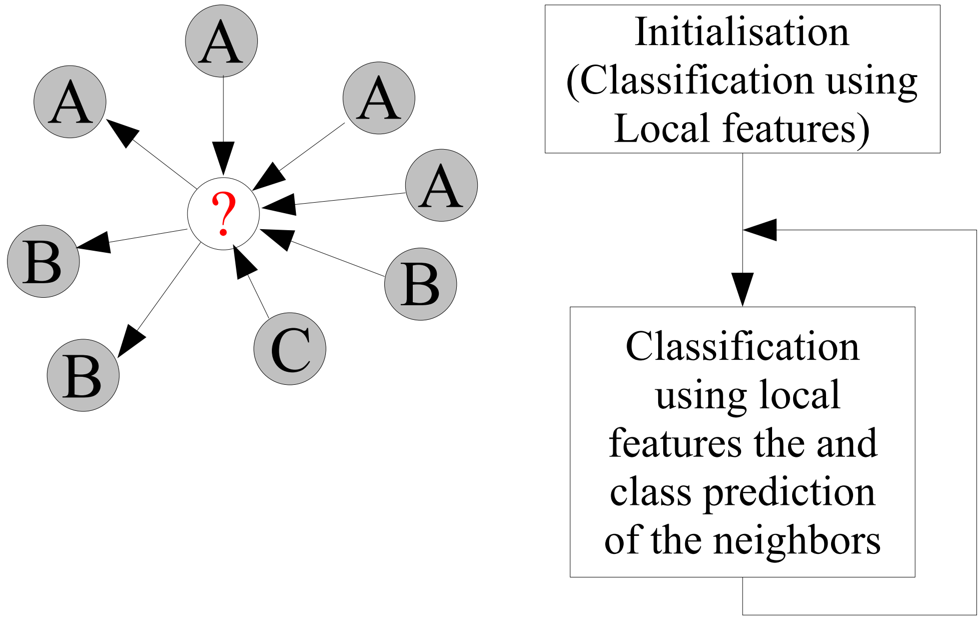

Figure 2.1: The iterative classifier of Chakrabarti

often lists links pointing to research departments, to its administration, to a description of the events occurring on the campus, to the athletics website, to the alumni students homepage or even to students facilities pages. But it links very rarely to the home page of an other university. Reciprocally, the home page of a university is rarely linked by the home page of an other university. Ensemble learning based on the category of the in or out-links would thus mislead if the problem is to identify home pages of universities. Co-citation links could be interesting in such cases but Getoor and Lu show in [12] that the average accuracy reached using co-citation links is generally worse than the one reached using in-links or out-links.

2.2 Link Mining

Lise Getoor lists different machine learning tasks based on link mining. She explains in [12] that link-based cluster analysis, record linkage and web pages classification can be improved with the help of link mining and she imagines new tasks that could be solved thanks to link information between the items of the dataset: Identifying the link type, predicting the link strength and determining the link cardinality.

Getoor and Lu [12] evaluate the improvements brought by link mining to web pages classification. Their model mines both local features on the documents and non-local statistical features computed from the category distribution of their neighbors. They distinguish three types of neighbors from the in-neighbors, the out-neighbors and the co-cited neighbors. An in-neighbor, a.k.a predecessor, is a web page that contains a link pointing to the target page, to the page to classify (target page). An out-neighbor is a web page that is linked by the target page. And the co-cited neighbors are web pages which have a common in-neighbor with the target page. We say that this common predecessor co-cites the two pages.

Getoor and Lu conjecture that statistics on the neighbors' categories distribution are as informative as the identity of the neighbors, which requires much more storing space. Of course, modifying the prediction for an example influences the predictions on its neighbors and the non-local features cannot be computed before the neighbors' category distribution has been predicted. Therefore, Getoor and Lu implement an iterative classifier (figure 2.2) on the model of Chakrabarti's that makes an initial prediction only based on the local features and then iteratively classifies the examples with the whole model, with both local and non-local features, until no prediction change happens.

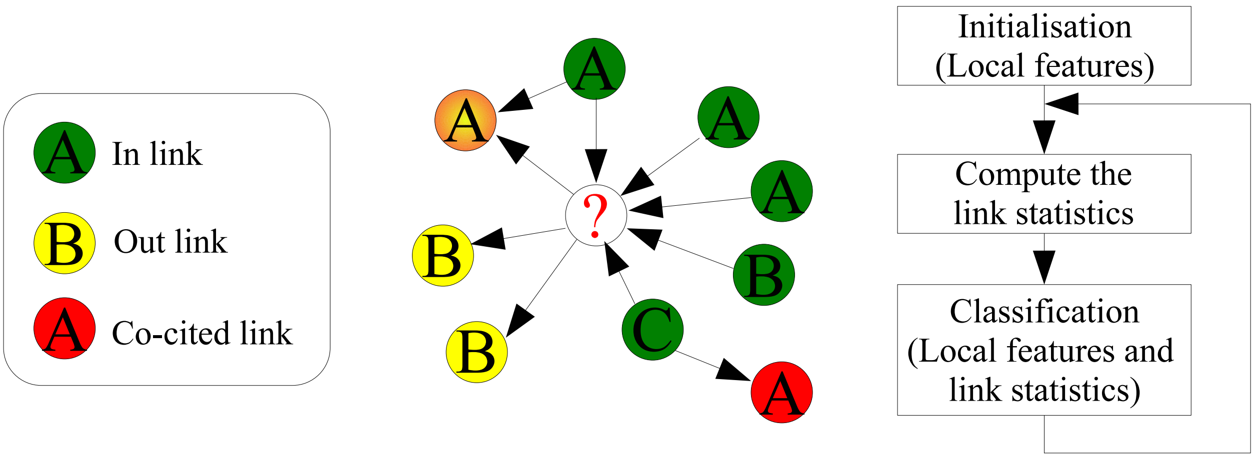

Figure 2.2: The iterative classifier of Getoor and Lu

They compare two classifier types: one flat model where the local features and the non-local ones were concatenated into a common vector. The local and non-local features are thus not distinguished. The second model is obtained by combining the predictions of both a classifier based on the local features and a classifier based on the non-local features. The flat model is outperformed by the second one, which confirms the results of Chakrabarti.

One interesting characteristic of their model is that instead of going with the majority prediction, they learn how the category distribution of the neighbors affects the prediction.

2.3 Web Page Categorization without the Web Page

As web indexers often collect several Uniform Resource Locators (URLs) while processing a single page and save them in order to classify them in a further iteration, the number of documents to be processed increases infinitely. Min-Yen Kan [10] studies for this purpose how a web page can be classified without retrieving its content. He bases his work on the URL and on informations mined in the hyperlink neighborhood that has already been retrieved by the indexer. His proposal is to firstly segment the URLs following the Uniform Resource Identifier protocol (scheme:://host/path-elements/document.extension) and to further segment wherever non-alphanumeric characters appear. The segments obtained are then expanded thanks to a title token based finite state transducer which associates the URL segments to their meaning mined in the titles of the documents processed in the training set. His tests show that an appropriate use of URL is three-fourth as effective as a text only based classifier. It outperforms also systems based on page title or anchor words. Unfortunately, adding the anchor texts reduces the performance and the URL only features fail to improve the performance of knowledge-rich classifiers.

Chapter 3

Our Hyperlink-based classifier

Our hyperlink-based classifier is an improvement of the model presented by Fürnkranz in [8] which mines not only local features but also non-local features on the predecessors. Our main addition is the use of the words neighboring the anchor description of the predecessors.

3.1 Introduction

Although the first attempts [5] to improve hypertext classification using informations mined in the hyperlink neighborhood resulted in an increase of the error rate, later works show that finer hypothesis can help to classify web pages. Some researchers have tested with success voting methods between the categories of the neighbors. Others [12] have shown that going with the majority prediction can lead to bad decisions while learning how the categories distribution of the neighbors affects the classification gives better results.

However, the categories of the neighbors can mislead in some cases. We believe that more than the categories of the neighbors, we should identify the category of each link. A predecessor may indeed contain a list of links pointing to pages of different categories. Intuitively, identifying each of those links separately is more accurate than trying to find a common category that would not be relevant for several links.

Furthermore, the classification methods based on the categories of the neighbors need iterative relaxation classifiers whose convergence is time expensive and uncertain. This is a handicap for search engines which are asked to have a short response time. As it only needs the features of the links to work, our model does not need any iterative classification process and is thus faster in classifying.

3.2 Overview

As for classical Text Classification, the features we mine on the documents are words. We test different heuristic patterns to target the words which give the most relevant information about each link, namely the *anchor description*, the *words neighboring the anchor*, the *headings structurally preceding the link*, *the heading of the list*, if the link is part of an HTML list and the *text of the document to classify*.

We evaluate various methods for combining these features. We process the different predecessors separately before computing a common classification or we use together the features of all the predecessors. We compare two multiclass problem binarizations: *One against all* and *Round robin*. We finally study how features mined from different sources shall be mutualized.

3.3 Feature patterns

In order to collect the information describing each link, we focus the mining of the features on precise spots specialized in retrieving one part of the classification clues. One last group of features, the whole text of the pages we want to classify, is mined in order to compare our model with a traditional text-based classifier.

PredLinkTags

The first spot is the link description, also named *anchor text* or *anchor propagation* in other studies like [4]. It consists of the text that occurs between the HTML Tags `<A HREF=...>` and `</A>` of the link pointing to the page to classify. If there are more than one link to the target page in a predecessor page, their descriptions are concatenated.

PredLinkHeadings

We use the clues highlighted by the HTML intern structure of the document. One of those clues is in a predecessor the headline of the paragraph that cites the target page. We mine in the *PredLinkHeadings* features group the words occurring in the headings *structurally* preceding the link in the predecessor. As three levels of headings exist in the HTML grammar (H1, H2 and H3), we concatenate in this group the last headline of each depth that occur before the link.

PredLinkParagraph

Simpler than the headline of the paragraph that cites the target page, the paragraph itself contains interesting words that describe the target page. We mine it in the features group *PredLinkParagraph*. We use the HTML tags `<P>` and `</P>` to find the borders the paragraph.

PredListHeadings

Sometimes, the link is part of a HTML list (tag `<UL>`). In this case, we store the preceding heading of each depth in the features group *PredListHeadings*

WordsAround

One difficulty with *PredLinkParagraph* is that the size of the paragraphs varies. The purity or the dilution of the clue features in the crowd of the words is not fixed. We circumvent this problem with the features group *WordsAround* where a fixed number of words neighboring the link are mined. The anchor description is excluded from *WordsAround*. This feature location is an important source of information for the links with an irrelevant anchor text like `click here` or `next page`.

OwnText

We mine the content of the target page in order to compare our model with traditional text-based classifiers.

3.4 Multiclass classification

Most of the classification problems are multiclass (figure 3.1) which means that the classifiers must decide between several categories. But almost all the machine learning algorithms are binary: They can only distinguish the positive class from the negative class. In order to solve multiclass problems with binary algorithms, we map them to equivalent binary classification problems. The two binarizations we implement are *One against all* and *Round Robin*. We illustrate them with a three-class classifier determining the language of an article: The document d_F to classify is in French and the possible categories are *English*, *German*, and *French*.

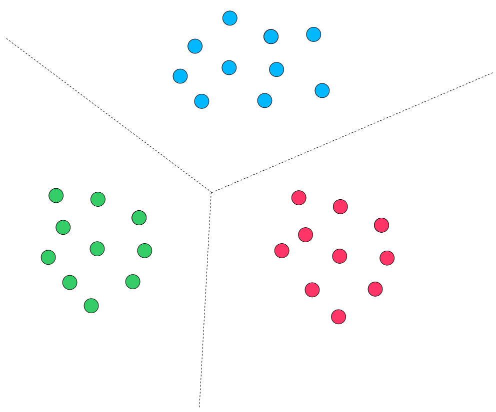

Figure 3.1: multiclass problem

3.4.1 One against all

Definition

The *one against all* binarization splits the n-class classification problem into n binary problems $\langle i \rangle$ as shown in figure 3.2 where the original class i is considered as the binary positive one and all the other original categories are viewed as a big negative category. With our example, the main problem is split into the three following binary problems and n binary classifiers are trained during the learning phase:

- $\langle 1 \rangle$ Is the text in *English* or not ?
- $\langle 2 \rangle$ Is the text in *German* or not ?
- $\langle 3 \rangle$ Is the text in *French* or not ?

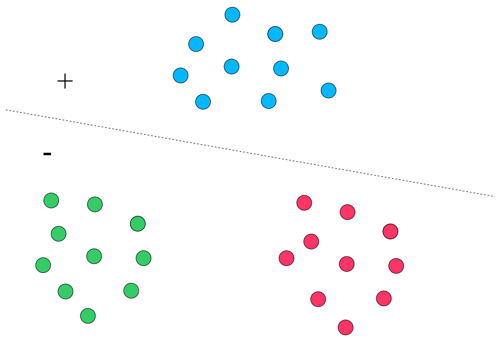

Figure 3.2: One against all

In the ideal case, the classification phase is obvious because all the classifiers answer negatively but the one corresponding to the correct class. The classifier $\langle 3 \rangle$ would answer *Yes*, and both classifiers $\langle 1 \rangle$ and $\langle 2 \rangle$ would answer *No*. However, real classifiers sometimes give erroneous results. This trouble is circumvented by a weighted vote.

$$score(C_i) = \frac{1}{n} \times (\quad c_i p_i - \sum_{i \neq j} c_j p_j)$$
where $p_k \in \{-1, 1\}$ is the prediction of the classifier $\langle k \rangle$ and c_k is its confidence rate.

Finally, the category chosen is the one that collects the maximum amount of points. In the following example, the category *French* has an average confidence of +39%, more than the categories *English* or *German*. Thereby, the document d_F is correctly classified as *French* even if the classifier $\langle 1 \rangle$ makes a false prediction.

One against all with Support Vector Machines

Support Vector Machines output a couple $(prediction, confidence)$. However, this confidence must be handled carefully because it depends not only on the probability of correctness estimated by the classifier but also on the minimum distance between an example

	Answer	*English*	*German*	*French*
Is the text in English ?	(1, 12%)	$1 \times 12\%$	$-1 \times 12\%$	$-1 \times 12\%$
Is the text in German ?	(-1, 55%)	$1 \times 55\%$	$-1 \times 55\%$	$1 \times 55\%$
Is the text in French ?	(1, 73%)	$-1 \times 73\%$	$-1 \times 73\%$	$1 \times 73\%$
Sum		$1 \times 10\%$	$-1 \times 47\%$	$1 \times 39\%$

Table 3.1: Example of One against all Vote

of the positive class and an example of the negative class. As the positive and the negative classes differ for each category specific binary classifier, the confidence rates of the classification of an example by different binary classifiers are influenced by two factors whose relative importance cannot be easily evaluated. Therefore, the confidence rates can unfortunately not be used to implement a weighted vote.

We give to each category specific classifier $(n-1)$ votes (where n is the number of categories). Each classifier distributes its voices over the categories according to its class prediction: All the $(n-1)$ voices for the positive category if the example is classified as positive, 1 voice for each of the $(n-1)$ categories of the negative class if the example is classified as negative.

In our illustration, if the classifier $\langle 1 \rangle$ (*Is the text in English ?*) misclassifies our document D_f, the result of the classification is

	Answer	*English*	*German*	*French*
Is the text in English ?	Yes	2	0	0
Is the text in German ?	No	1	0	1
Is the text in French ?	Yes	0	0	2
Sum		**3**	0	**3**

Table 3.2: Example of One against all Vote with Support Vector Machines

Hence the document d_F is either in *English* or in *French.* This case of undecidability is not rare and is solved by choosing the most populated category between the categories that got the best score.

3.4.2 Round robin

The *Round Robin* or *pairwise* class binarization [2], figure 3.3, transforms one n-class problem into $\frac{n(n-1)}{2}$ binary problems $\langle i, j \rangle$: One for each pair of classes $\{i, j\}$, with $i, j \in \{1..n\}, i \neq j$. The binary classifier for the problem $\langle i, j \rangle$ is trained with the examples of the classes i and j, whereas the examples of the classes $k \neq i, j$ are ignored at this stage.

As with *One against all*, the confidence rate of Support Vector Machines cannot be used for this purpose. That is why we give for each binary classification 1 point to the winner category and -1 point to the looser one. If several categories remain possible, we choose the most populated between the categories that got the best score.

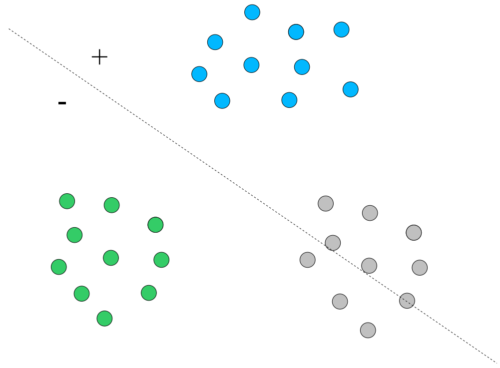

Figure 3.3: Round Robin

	Answer	*English*	*German*	*French*
Is it English or German ?	English	1	-1	0
Is it English or French ?	French	-1	0	1
Is it German or French ?	French	0	-1	1
Sum		0	-2	**2**

Table 3.3: Example of Round Robin Vote with Support Vector Machines

3.5 Learning from many predecessors

In traditional classification problems, one set of features is associated with each member of the dataset. The particularity of our approach is to handle an ensemble of feature sets for each document. Each predecessor of the document to classify brings its own feature set. The challenge is that the number of predecessors varies and that there is no clear order between them.

3.5.1 Meta Predecessor

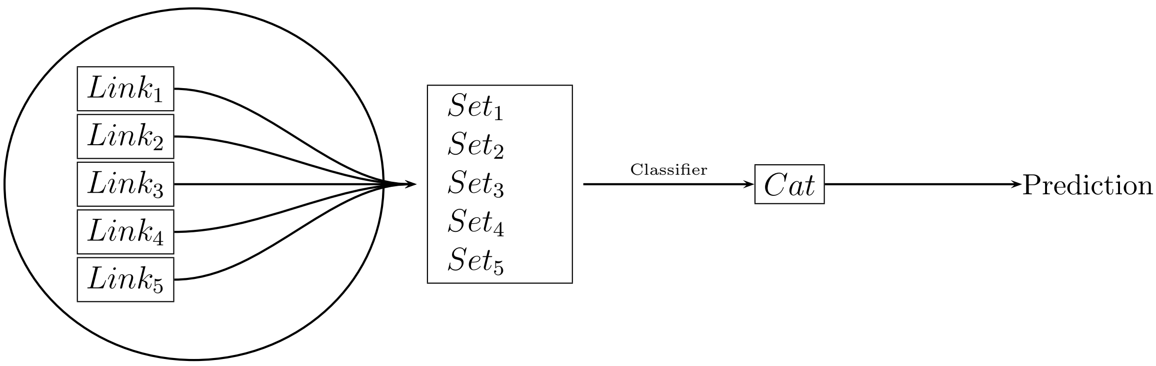

Figure 3.4: Meta predecessor

The first solution (figure 3.4) we test is to create a *meta predecessor* which aggregates all the features mined on the different predecessors as shown in figure 3.5 with two predecessors and the anchor description as unique feature mined.

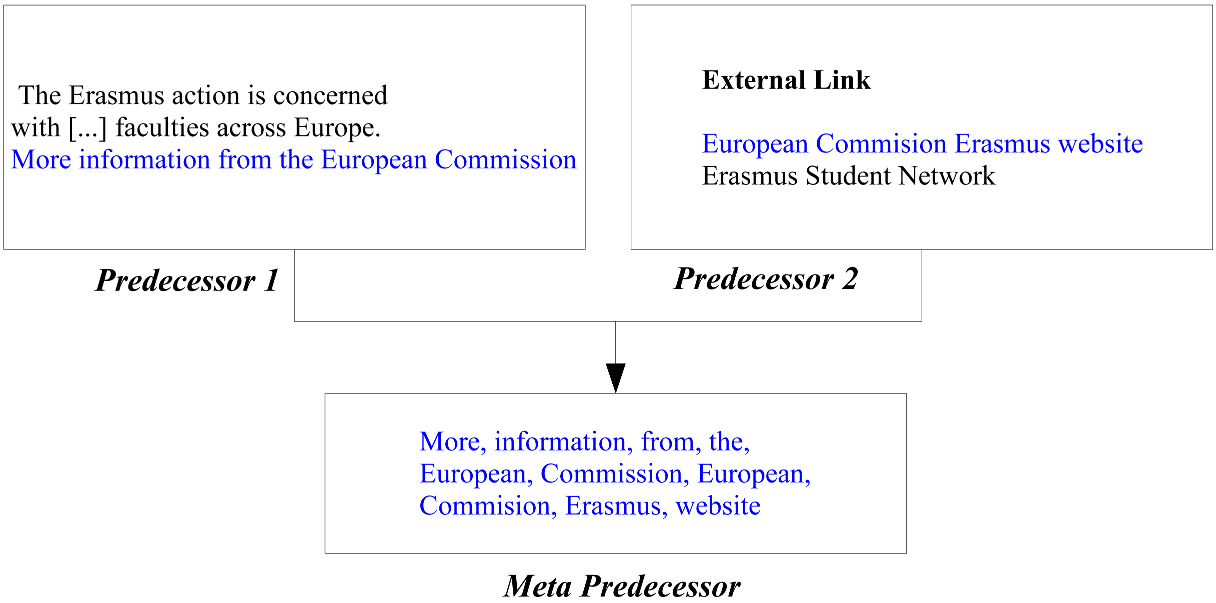

Figure 3.5: Meta Predecessor

3.5.2 Ensembles

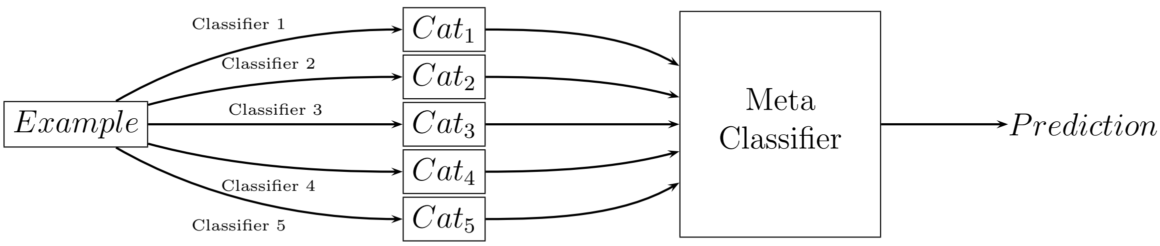

Figure 3.6: Meta Learning

Machine learning proposes different classifiers. The best-known are Support Vector Machines, Decision Trees, Neural Networks or Naive Bayes models. There is no total order between the classifiers. Some problems are better solved by Naive Bayes classifiers than by Decision Trees. But other problems are better solved by Decision Trees than by Naive Bayes classifiers. The order depends on the problem to solve. Sometimes, even for a given problem, the ordering is different for each class. SVMs could for example outperform Decision Trees for a given category of data while the relative performance of these algorithms would be inverted for an other category. Meta Learning [6] (figure

3.6) has been developed for this purpose. Its principle is quite simple: As there is no best classification algorithm, a selection of different methods are run concurrently. A final prediction is then computed by a meta classifier regarding the results of all the algorithms selected.

3.5.3 Hyperlink Ensembles

In hypertext classification, the problem is slightly different. The classification algorithm is usually chosen in advance, but what has to be determined on the fly is the relative importance that should be granted to each neighbor of the target page. Despites obvious similarities, stacking can't be directly applied to hypertext classification because the number of links varies and because there is no clear order between them. If traditional stacking can't be implemented for our study, its ground idea remains interesting and can be extended for our purpose.

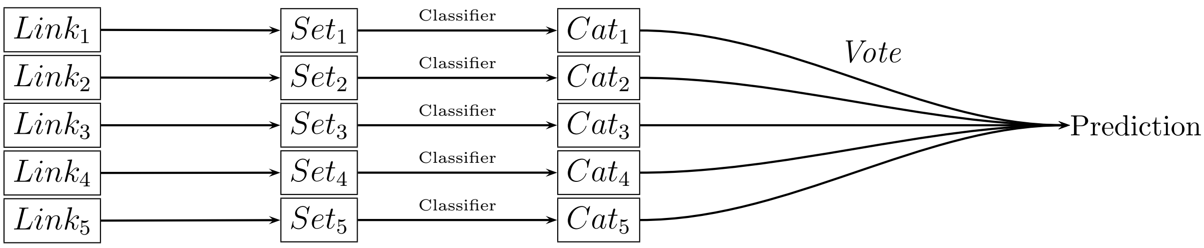

Figure 3.7: Hyperlink Ensembles

As in traditional stacking, each link is considered in hyperlink ensembles (figure 3.7) as an entity which is classified independently. A pair ($prediction, confidence$) is computed for each link. The set of these pairs form the *hyperlink prediction ensemble* which feeds an ensemble meta classifier that computes a final prediction and confidence. This meta classifier is usually a heuristic like voting, weighted sum or the prediction with the maximum confidence level. It can be a meta learner based on statistics computed on the hyperlink prediction ensemble. Examples for these statistics are tuples representing the distribution of the categories in the ensemble or indicating the presence or absence of each category in the ensemble[12]. In our study, we implement a non weighted vote.

3.6 Mutualizing the feature patterns

Once features have been mined thanks to different patterns, we must tie them together. The goal is to create a *meta feature set* which contains all the features mined by the different patterns and which will be the input of both learning and classification algorithms. This is not obvious because some patterns are very permissive and collect many spurious words. Other ones are very selective and the few words mined are very reliable. We compare two solutions, namely Merging and Tagging.

3.6.1 Merging

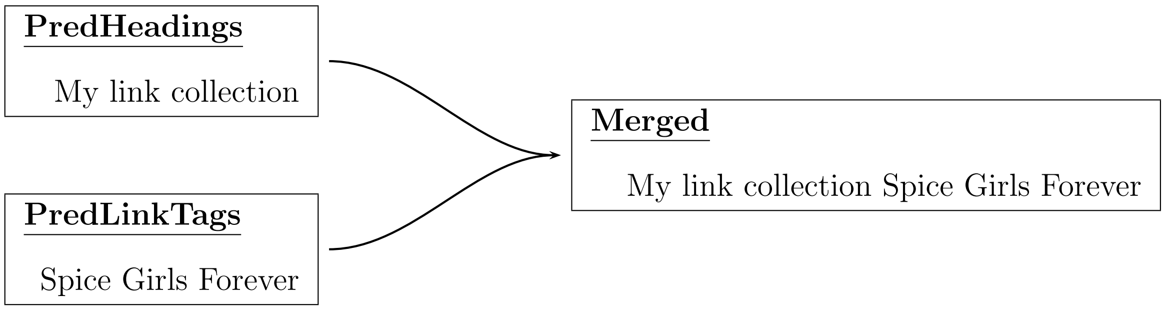

Figure 3.8: Merging

The first solution (figure 3.8) is merging all the words with the same weight for all the mining methods, which has the inconvenient to dilute strong selected words in a flow of noisy features.

3.6.2 Tagging

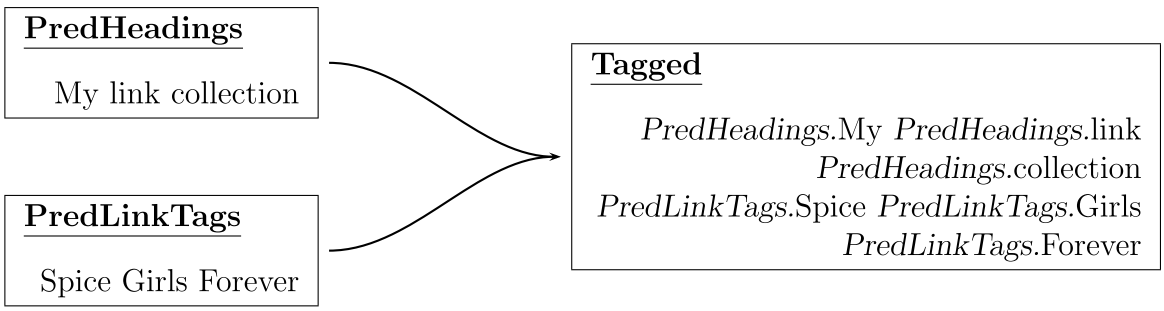

Figure 3.9: Tagging

The second one, Tagging (figure 3.9), is to consider identical words mined by two methods as two different features. The major problem with that solution is a loss of redundancy.

In order to make a distinction between identical words mined by different patterns, we tag each feature with its pattern name. For example, the word `spice` mined by the method `PredLinkTags` is stored under the feature name `PredLinkTags.spice`. For the both solutions, the bags of words of the features mined by the different patterns are put together in a common bag of words.

Chapter 4

Implementation of our model

In this chapter, we present the two benchmark collections used for evaluating our model, an overview of the Support Vector Machines, the classification algorithm chosen. We finally describe the preprocessing applied before the classification and we explain how the features are mined.

4.1 The benchmark collections

The datasets we use for evaluating the viability of our approach are two labeled web pages collections with a more or less strong hyperlink connectivity. The first one, *Allesklar*, has been specifically collected for this study. It is strongly connected and a majority of its web pages has more than 10 predecessors what permits a full use of ensemble classifiers. The other one, *WebKB*, has been collected for other purposes and has already been used as benchmark collection for other text classification algorithms by different researchers [8, 18]. As WebKB has not been meant for hyperlink ensemble classifiers, it is weaklier connected than the Allesklar dataset.

4.1.1 The Allesklar dataset

Alleskar (http://www.allesklar.de) is a German generic web directory referencing about 3 million of German web sites. Its tree organization begins with 16 main category roots, each one containing between 30 000 and 1 000 000 of sites. The nodes of the tree are as specific categories as the node is deep. We chose 5 main categories, namely *Arbeit und Beruf (Work and Jobs), Bildung und Wissenschaft (Education and Science), Freizeit und Lifestyle (Hobbies and Lifestyle), Gesellschaft und Politik (Society and Politics) and Immobilien und Wohnen (Accommodation).* They are rather equally distributed as shown in table 4.1.

We crawled each selected category with a breadth-first traversal in order to collect pages covering the whole category. We looked for hyperlink predecessors for each of these pages thanks to the Altavista link request (for example, the request `link:europa.eu.int`

Category	Examples
Arbeit&Beruf	578
Bildung&Wissenschaft	809
Freizeit&Lifestyle	752
Gesellschaft&Politik	833
Immobilien&Wohnen	793

Table 4.1: Category distribution for Allesklar

retrieves all the web sites containing a link to the Web portal of the European Commission).

We looked for up to 25 predecessors per example, but we couldn't always find as many predecessors and the predecessors referenced by altavista were not always reachable. In figure 4.1 we show for each category the distribution of the cardinalities of the subsets of the benchmark collection that have a given in-degree. Only a few part of the examples have no predecessor and a large part of them has more than 10 predecessors. There is no important difference between the categories from this point of view. Only the distribution of *Immobilien und Wohnen* is slightly shifted to fewer predecessors.

In order to shorten the response time and accelerate the crawling, we implemented a proxy which avoided multiple downloads of a common predecessor of different members of the Alleskl̈ar directory. We saved the elements of the dataset in separate files whose name is their URL slightly modified to make it compatible with the Unix file naming constraints. We added two more files to the dataset: `_Classification`, which lists the categories of the files and `_Predecessors` which saves the hyperlink graph structure of the dataset.

The file `_Classification` (table 4.2) describes one document per line. Each record is composed of three fields separated by commas: The Unix filename, the category and the URL of the document.

Each line of the file `_Predecessors` (table 4.3) lists the in-links of a document. Each record is composed of the Unix filename of the document, a colon, and the list of its predecessors separated by semicolons. In this extract, the lines have been truncated. The actual average number of predecessors (in-degree) in the Alleskl̈ar dataset is 14.70

aaa-botzke.de	, Immobilien-Wohnen	, aaa-botzke.de
aaonline.dkf.de^bb^p109.htm	, Arbeit-Beruf	, aaonline.dkf.de/bb/p109.htm
abb-angermuende.de	, Immobilien-Wohnen	, abb-angermuende.de
action5.toplink.de	, Gesellschaft-Politik	, action5.toplink.de
agenturohnegrenzen.de	, Freizeit-Lifestyle	, agenturohnegrenzen.de
aib-backnang.de	, Arbeit-Beruf	, aib-backnang.de
akzente-zuelpich.de	, Immobilien-Wohnen	, akzente-zuelpich.de
allschutz.de	, Immobilien-Wohnen	, allschutz.de
anahato.bei.t-online.de	, Freizeit-Lifestyle	, anahato.bei.t-online.de
anderswelt.com^kreiszeit	, Freizeit-Lifestyle	, anderswelt.com/kreiszeit

Table 4.2: Sample part of the file _Classification

from aaonline.dkf.de^bb^p109.htm : www.ralf-bales.de^gesamt.htm ; www.open-skies.org^hopepage^links.html ; ...
from berufenet.arbeitsamt.de : www.studienwahl.de^fmg.htm ; www.was-werden.de ; ...
from home.degnet.de^koller_stefan^lyrics^ly_start.htm : lyrics.berger-rangers.de ; elcapitano.berger-rangers.de ; ...
from home.t-online.de^home^schmidt.re : www.lyrik.ch^lyrik^links.htm ; www.lyrik.de ; www.haikulinde.de^links.htm ; ...

Table 4.3: Sample part of the file _Predecessors

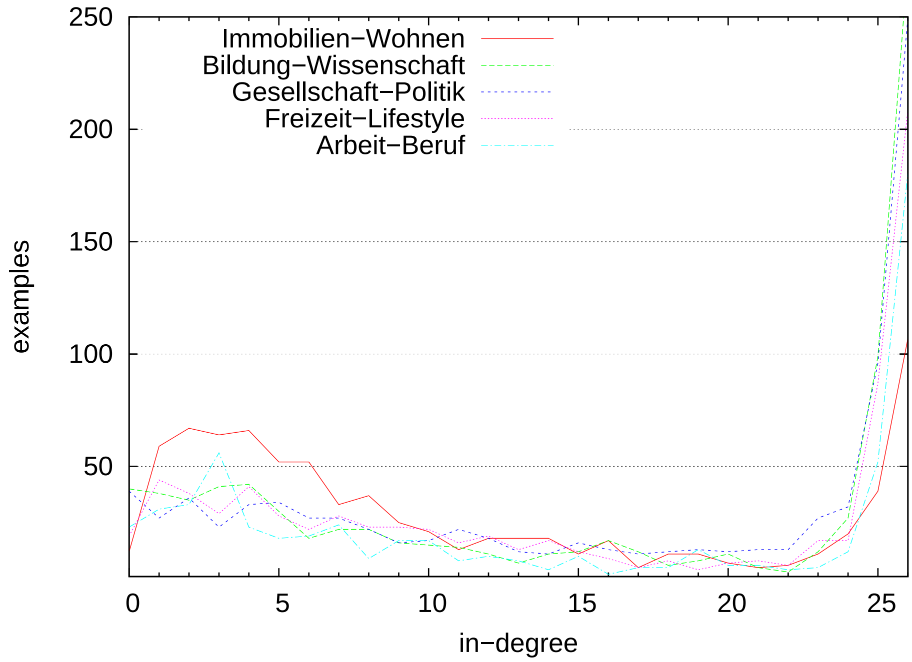

Figure 4.1: Distribution of the documents of Alleskla r

4.1.2 The WebKB dataset

The WebKB dataset is a collection of web pages coming from the science departments of four main universities: *Cornell*, *Texas*, *Washington* and *Wisconsin*. One fifth group of pages named *misc* has been collected from various other universities. These pages are classified under seven categories: *course*, *department*, *faculty*, *project*, *staff*, *student* and *other*. The WebKB dataset is not equally distributed (table 4.4): More than 45% of the examples are concentrated in the hold all category *other* while only 1.5% of the examples are classified as *staff* pages, which makes this dataset particularly difficult to classify.

This dataset was already collected, but we still had to discover its hyperlink graph. We made this by parsing each member of the dataset, looking if the targets of the hyperlinks were members of the dataset. As there are different ways to write a URL, two URLs cannot be compared character by character: The protocol descriptor *(ftp://, http://)* may be written or not, the paths may be relative or absolute, some servers are case sensitive

category	Examples
other	3756
student	1639
faculty	1121
course	926
project	506
department	181
staff	135

Table 4.4: Category distribution for WebKB

but not all, some URLs may contain PhP variables and their values. Thereby we implemented a function based on the Perl module URI::URL which simplifies the URLs into a homogeneous format.

We could then explore the hyperlink graph of the datasets by rewriting each link target (stored in a `<A HREF=...>` HTML tag) in the pages into the simplified format and by looking if the targets were present in the dataset. Statistics about WebKB's graph structure are shown in figure 4.2.

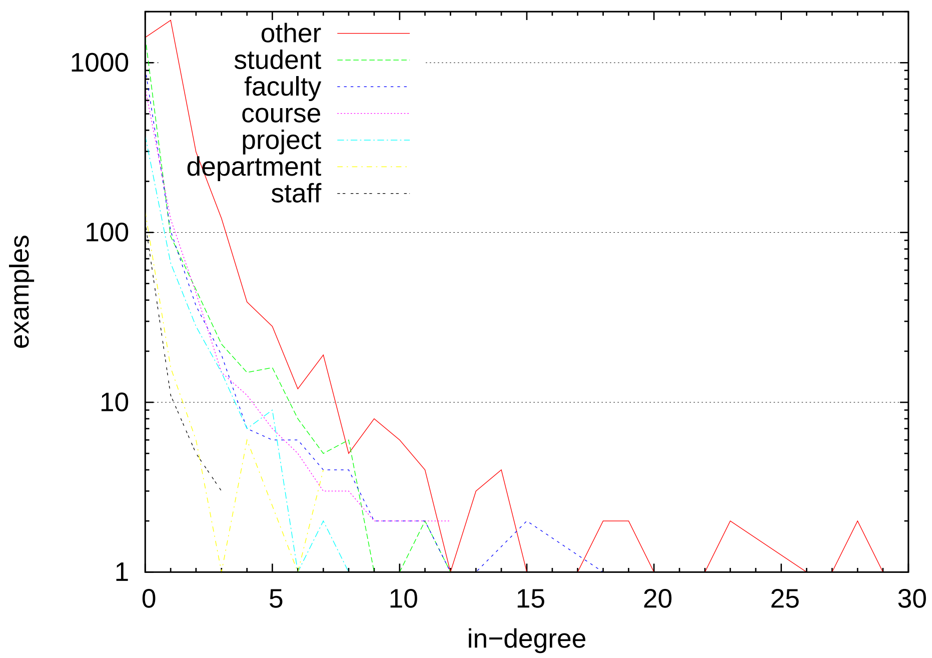

Figure 4.2: Distribution of the documents of WebKB

As the dataset hadn't been built for a hyperlink ensemble study, its graph structure is dramatically weaker connected than that of Alleskla r. No predecessor could be found for 5082 pages of the dataset among 8276 and only 67 pages own more than 10 predecessors.

4.2 Linear Support Vector Machines

4.2.1 Overview

The key trick of linear Support Vector Machines is to handle the feature values of the examples like coordinates in a vector space. As a similarity between two documents implies a proximity between their features values, the documents of the same class aggregate in clusters. The goal of Support Vector Machines is then to find a surface that separates the points of the two classes as good as possible. In the case of linear Support Vector Machines, this surface is an hyperplane (see figure 4.3).

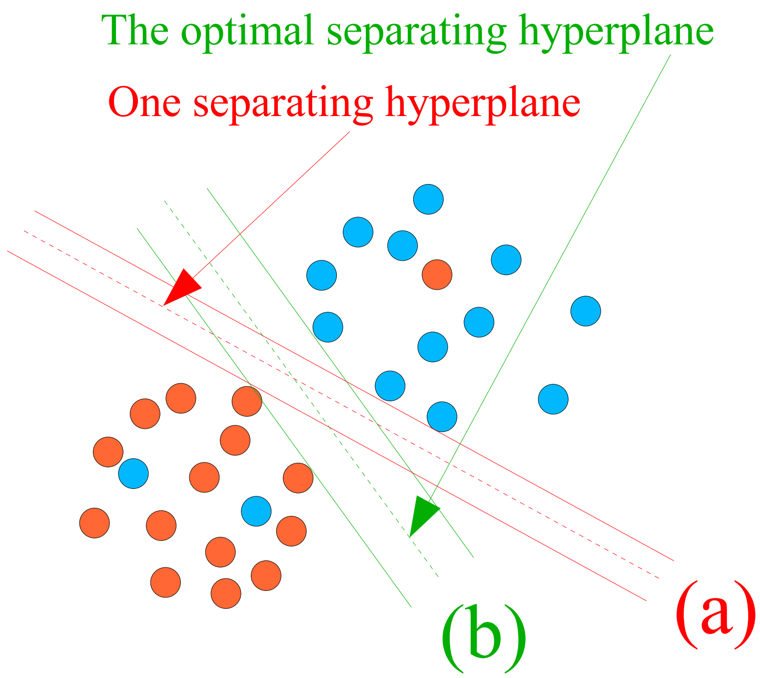

Figure 4.3: The optimal hyperplane is (b) while the separating hyperplane (a) has a narrower error margin

4.2.2 The Linear Support Vector Algorithm

Each one of the n examples δ_i of the training set $\mathscr{T}$ is represented by its vector $\vec{x_i} \in \mathcal{R}^d$. In the case of text classification, the coordinates of this vector are the occurrences of the words mined. d is the dimensionality of the problem. In the case of text classification, d is the number of different words mined. For example, the figure 4.4 shows the vector for the phrase *A gift is a gift*. In this example, $d = 3$ (`a, gift, is`).

Each example d_i is labeled by $y_i \in \{-1, 1\}$ ($y_i = 1$ if the example is in the positive class, $y_i = -1$ otherwise). A hyperplane whose coordinates are $(\vec{w}, b) \in \mathcal{R}^d \times \mathcal{R}$ separates the two classes if the inequation 4.1 is verified.

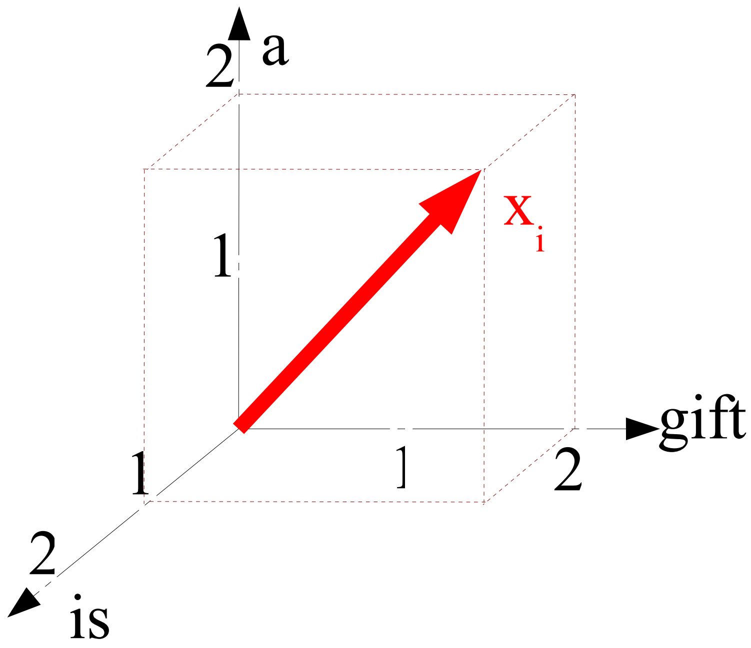

Figure 4.4: Vector for the phrase A gift is a gift

$$\forall i \in [1, n], y_i(\vec{w} \cdot \vec{x_i} + b) \geq 0 \tag{4.1}$$

We suppose that the dataset is linearly separable and thereby that there exists such hyperplanes. As a hyperplane is determined only by the direction of $\vec{w}$ and by the threshold b but not by the norm $\parallel \vec{w} \parallel$, we can without lost of generality rescale the pair $(\vec{w}, b)$ into $(\vec{w_0}, b')$ so that the distance of the closest document, say δ_j, to the hyperplane equals $\frac{1}{\|\vec{w_0}\|}$

The signed distance d_i of a document δ_i to the hyperplane is given by

$$d_i = \frac{\vec{w_0} \cdot \vec{x_i} + b'}{\parallel \vec{w_0} \parallel} \tag{4.2}$$

And thus, with 4.1 and 4.2,

$$\forall \delta_i \in \mathscr{T}, y_i d_i \geq \frac{1}{\parallel \vec{w_0} \parallel} \tag{4.3}$$

The optimal hyperplane is the separating hyperplane with the biggest error margin. In other words, it is the one whose distance to the closest points (support vectors) of $\mathscr{T}$ is maximum. Thereby, the goal is to maximize $\frac{1}{\|\vec{w_0}\|}$

Maximizing $\frac{1}{\|\vec{w_0}\|}$ is equivalent to minimizing $\parallel \vec{w_0} \parallel$ and thus to minimizing $\frac{1}{2}\, \vec{w_0} \cdot \vec{w_0}$.

However, a dataset is rarely totally linearly separable. It is thus necessary to examine the examples and to determine the weight that should be given to each one for the classification. The remark that motivates this weight distribution is that the optimality of the hyperplane depends more on the points disposed on the border between the two classes than on points disposed far away from this border and that would be correctly classified even is the hyperplane were slightly moved. Furthermore, a single positive example at the middle of a group of negative examples should be discarded so that a separating hyperplane may be found.

Determining which weight shall be granted to each example is a difficult optimization problem that has been solved with the help of the successive works of mathematicians. Pierre de Fermat published in 1629 the first method to find the minimums and the maximums of a function. This method was adapted by Lagrange in 1797 to mechanical optimization problems, and Kuhn and Tucker extended the Lagrangian theory in 1951 so that not only equality constraints but also inequality constraints can be taken into account in the optimization.

The problem of minimizing $\frac{1}{2}\vec{w_0}\cdot\vec{w_0}$ subject to the correct classification constraint $\forall i \in [1,n], y_i(\vec{w_0}\cdot\vec{x_i}+b) \geq 1$ becomes with the relative weight α_i granted to each example δ_i the problem of finding the saddle point (figure 4.5) of the function L.

$$L = \frac{1}{2}\vec{w}\cdot\vec{w} - \sum_{i=1}^{n}\alpha_i(y_i(\vec{w}\cdot\vec{x_i}+b)-1) \tag{4.4}$$

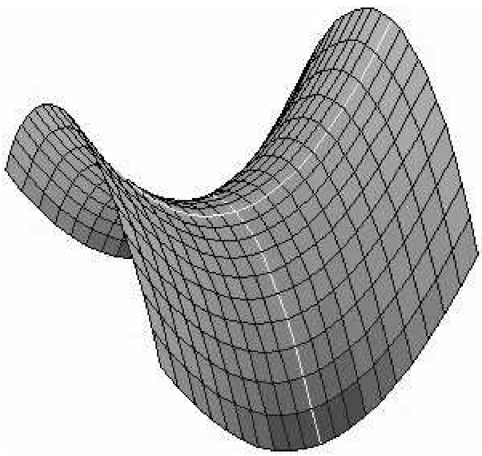

Figure 4.5: Saddle point

At the saddle point,

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} y_i\alpha_i = 0 \tag{4.5}$$

$$\frac{\partial L}{\partial \vec{w}} = \vec{w} - \sum_{i=1}^{n} y_i\alpha_i\vec{x_i} = \vec{0} \tag{4.6}$$

with

$$\frac{\partial L}{\partial \vec{w}} = (\frac{\partial L}{w_1}, \frac{\partial L}{w_2}, ..., \frac{\partial L}{w_d}) \tag{4.7}$$

The hyperplane coordinates $(\vec{w}, b)$ are thus given by

$$\begin{cases} \vec{w} &= \sum_{i=1}^{n} y_i \alpha_i \vec{x_i} \\ b &= ArgMax(\sum_{i=1}^{n} \alpha_i y_i (\vec{w} \cdot \vec{x_i} - 1)) \end{cases}$$

Once the hyperplane has been determined, the classification phase is an easy task: It consists of looking at which side of the hyperplane is the document to classify $\vec{d}$.

$$\vec{w} \cdot \vec{d} + b \begin{cases} \geq & +1 & \vec{d} \text{ is positive} \\ \in & [0;1[& \vec{d} \text{ is probably positive but in the error margin} \\ \in &]-1;0[& \vec{d} \text{ is probably negative but in the error margin} \\ \leq & -1 & \vec{d} \text{ is negative} \end{cases}$$

Hence, the classification is given by the *Decision function* $D(\vec{d})$

$$D(\vec{d}) = sign(\vec{w} \cdot \vec{d} + b)$$

4.2.3 Comparison with other classification algorithms

Fabrizio Sebastiani ranks various classification algorithms in [16]. He bases its conclusions on the works of Schütze[14], Schapire[13], Dumais[7] and Yang[17]. Support vector machines appear to be with boosting-based classifier committees in the top performing group. Then come neural networks and on-line linear classifiers and the least performing ones are Rocchio classifiers and naive Bayes classifiers.

4.3 Preprocessing

All the documents are preprocessed in the following way: The HTML tags are removed and the text is lower cased. The diacritic signs (accents, cedilla, Spanish tildes) are removed and the German characters ß, ä, ö and ü are replaced by ss, ae, oe and ue. Each remaining non alphanumeric character is then replaced by an underscore and the numbers of one or more digits by a single D. As the text has been lowercased before, there is no risk of confusion between the letter d and a number represented by a D. If several successive underscores are found, they are reduced to a single underscore. The underscores occurring at the beginning or at the end of a word are removed so that words framed by parenthesis or quotes or followed by a point or a coma are considered like those that are framed by spaces. The remaining words are finally filtered by a German and English *stop words* list (annexe A).

4.4 Mining of the features

We implement the features mining using XPath [1] structural patterns on the Document Object Model (DOM) representation of the documents. XPath is a language for navigating through elements and attributes on an XML document. It uses path expressions to select nodes or node-sets in an XML document. These path expressions are similar to those used for locating files in a filesystem.

In order to make the web pages browsable with XPath expressions, we firstly translate them from HTML format into XHTML format with the help of Tidy. We encountered a problem by this step because some HTML pages contain many syntax errors. Tidy cannot understand them all and can thus not output the XHTML translation of all the documents. We circumvent this difficulty by mining the basic features (text of the target page) on the HTML page before the Tidy treatment and the complex features (anchor description, headings, ...) after the construction of the DOM tree by Tidy.

Table 4.5 lists the XPath expressions we use to extract the features from the predecessors of the target document. In these expressions, Target_SURL is replaced by the simplified form of the URL of the target page. The common prefix of the expressions (//a[\@href='Target_SURL']) is divided into three parts.

Expression	Meaning
//	*target all the*
a	*anchor tags*
[\@href='Target_SURL']	*whose attribute* href *is set to* Target_SURL

Hence, the result of the PredLinkTags request is the concatenation of the segments of the XHTML file that occurre between the HTML tags <A HREF=Target_SURL> and </A>, tags included. The other requests are simple extensions of the PredLinkTags request. Once the anchor tag of the links is localized, PredLinkParagrah looks for its last ancestor of type Paragraph. PredLinkHeadings looks for the last occurrence of each heading level before the link, and PredListHeadings looks for the last occurrence of each heading level before the beginning of the list.

Table 4.5: XPath expressions

<table>
<tr><td>PredLinkTags</td><td>//a[\@href='Target_SURL']</td></tr>
<tr><td>PredLinkParagraph</td><td>//a[\@href='Target_SURL']/ancestor::p[last()]</td></tr>
<tr><td>PredLinkHeadings</td><td>//a[\@href='Target_SURL']/preceding::h1[last()]
| //a[\@href='Target_SURL']/preceding::h2[last()]
| //a[\@href='Target_SURL']/preceding::h3[last()]</td></tr>
<tr><td>PredListHeadings</td><td>//a[\@href='Target_SURL']/ancestor
::ul/preceding::h1[last()]
| //a[\@href='Target_SURL']/ancestor
::ul/preceding::h2[last()]
| //a[\@href='Target_SURL']/ancestor
::ul/preceding::h3[last()]</td></tr>
</table>

Chapter 5

Experimental Set Up

We describe in this chapter the cross splitting algorithm implemented for our hyperlink-based classifier. We present a reflection about the size of the learning sets and we motivate the choice of using a document frequency based dimensionality reduction. We finally give precision about the experimental environment.

5.1 Cross validation

Cross splitting is a sensible point of the classifier evaluation. If a *cross validation* is often used in order to shorten the computing time which would be needed for a *leave one out* validation, this random method introduces a bias and is difficult to reproduce exactly. We will also describe precisely the stratified splitting method chosen so that out experiments are reproducible.

We set a counter for each category and distribute the examples one after the other in the order of the file `_Classification`. The first example of a given category is added to the test set of the first fold and to the training sets of all the other folds. The second example of that category is added to the test set of the second fold and to the training sets of the other folds etc...

With n folds, the distribution criterion for the e^{th} example of a given category is: $\forall f \in [1,n]$, if $e \equiv f\,[n]$, the example is added to the test set of the fold number f. Otherwise, it is added to the training set of this fold.

This splitting method respects the splitting law saying that each example is in one and exactly one test set amid all the folds.

$$\begin{cases} \forall\ e\ example, \exists\ f fold, e \in f.test \\ \forall f_1, f_2\ folds, f_1 \neq f_2 \Rightarrow f_1.test \cap f_2.test = \emptyset \end{cases}$$

5.2 Size of the training set

One challenge of the learning phase is to correctly choose the size of the training set. Too little, the set would be too weakly correlated so that the inductive learning process may extract the characteristics of the categories. Too big, the learning time would increase without a corresponding increase in the effectiveness of the classifier. As the accuracy can't grow indefinitely, we guess that there exists a given number of training examples n that is fruitless to exceed.

In order to determinate this threshold, we train our hyperlink based classifier with a growing number of learning examples, and we test it on a fixed test set (figures and 5.1 5.2)

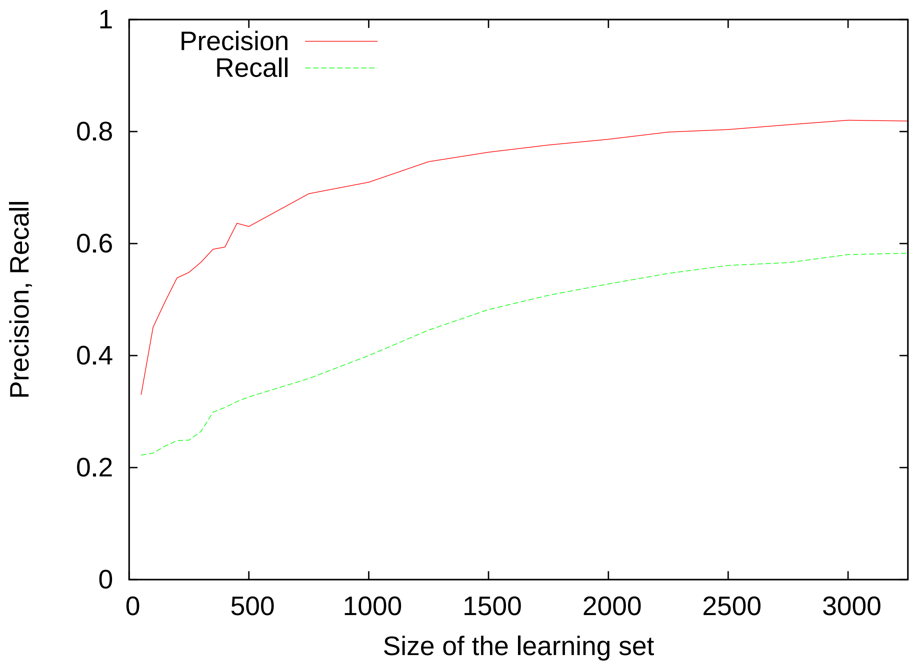

Figure 5.1: Size of the training set for Allesklar

This experiment confirms that adding new training examples improves the accuracy of the classifier. This result is common to the precision and the recall of both WebKB and Allesklar datasets. The threshold is reached with 2500 examples on WebKB. Both precision and recall clearly grow before this value and stagnate after. Unfortunately, the number of examples in the Allesklar dataset is too little so that we can determinate its threshold value.

5.3 Dimensionality reduction

The classification algorithms use the redundancy of the information to extract from the training set statistical rules that describe the categories. Rare words are hardly seen by the classifiers and are therefore not used in the classification rules. Thereby, filtering them out of the training set does not hinder the learning phase. On the contrary, it reduces the dimensionality of the problem, what makes the classification easier. Fabrizio Sebastiani

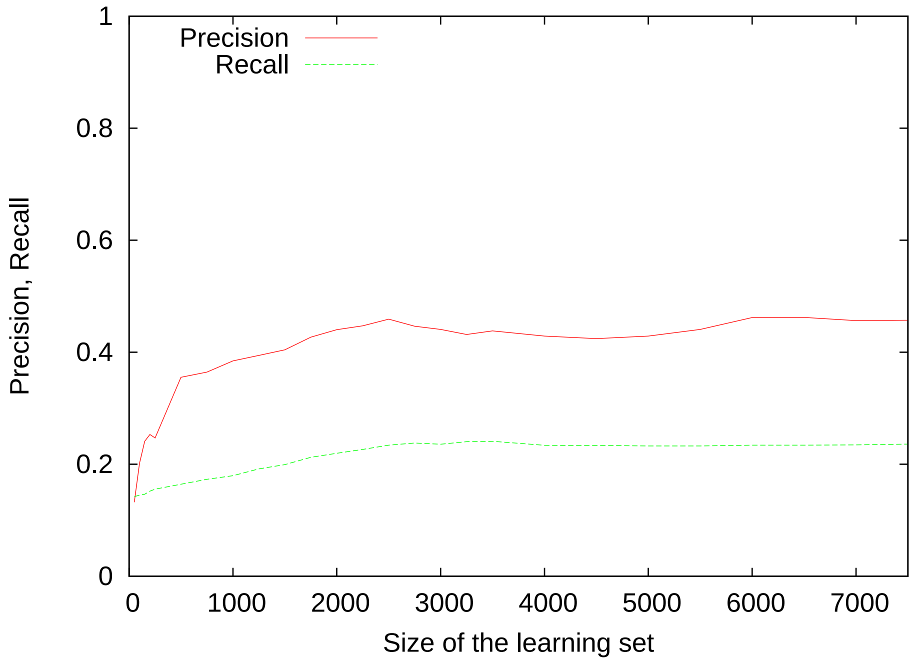

Figure 5.2: Size of the training set for WebKB

collects results in [16] on dimensionality reduction based on the document frequency of the features. It appears that reducing the dimensionality by a factor of 10 does not hinder the effectiveness of the classifiers while a factor of 100 brings about just a small lost. For our classifiers, we chose a document frequency based dimensionality reduction by a factor 10.

5.4 Experimental environment

We lead the experiments on a bi processor (2 AMD Opteron, 2.4Ghz) Linux station running the kernel 2.6.9. The support vector machine algorithm we use is SVM-light V6.01 written by Thorsten Joachims. We write the scripts which process the data before and after SVM-light with Perl v5.8.5. The version of Tidy used for the features mining is the one released on the first of September 2004.

Chapter 6

Results

In this last chapter, we explain the methods we use for evaluating our different classifiers, we propose an evaluation of the different sources of features and we explain the heavy points and the disadvantages of the different classification techniques tested and we finally present detailed results about our best hyperlink-based classifier.

6.1 Evaluation

6.1.1 Evaluation of a classifier

Evaluation functions: accuracy, precision, recall and F_β

Several evaluation functions have been imagined for measuring the effectiveness of text classification methods. The most common ones are *accuracy*, *precision*, *recall* and F_β. Those functions are computed from the confusion matrix.

Category c_i	Classified as positive	Classified as negative
Is positive	a	b
Is negative	c	d

Accuracy The accuracy is the probability that a document is correctly classified. This measure is estimated by the statistical function $A = \frac{a+d}{a+b+c+d}$. Accuracy can however mislead in the case of a multiclass problem.

Precision The precision is the fraction of retrieved documents that are relevant. It is measured by the function $\pi = \frac{a}{a+c}$. This is the most important evaluator for this study since the web users don't await the search engines to give them an exhaustive list of the pages treating a particular subject. But they want that the pages proposed are relevant.

Recall The recall is the fraction of relevant documents that are retrieved. It is measured by the function $\rho = \frac{a}{a+b}$. Recall can't be used alone to evaluate a classifier because it can be artificially increased to the detriment of precision by classifying every document as positive.

$\mathbf{F}_\beta$ The function $F_\beta = \frac{(\beta^2+1)\pi\rho}{\beta^2\pi+\rho}$ is a weighted compromise between precision and recall.

$$\lim_{\beta\to\infty}(F_\beta) = \rho$$

and

$$\lim_{\beta\to 0}(F_\beta) = \pi$$

The typical value for β is 1, which give an equal weight to π and ρ.

$$F_1 = \frac{\rho\pi}{\rho+\pi}$$

Micro averaging and Macro Averaging

When the examples are distributed between more than two categories, there are two ways to compute precision and recall, and also F_β. The first one, called *micro averaging*, consists of calculating the 2×2 confusion matrix of each category and of summing them in a global 2×2 confusion matrix from which the evaluation measures are computed as explained in section 6.1.1. *Macro averaging* computes the evaluation measure for each individual category and averages them over all categories. *Micro averaging* emphasizes the most populated categories whereas *macro averaging* emphasizes the least populated ones.

Cross-Validation

A similar generalization must be done when cross validation is implemented. The evaluation functions as defined before are computed on a each fold. We make a micro-averaging-like computation of F_1: We add the confusion matrices of the different fold tests and calculate the micro-average values of recall, precision and then F_1 on this global matrix.

Choice of the evaluation function

Most of the Text Classification problems consist of finding all the relevant documents corresponding to a query. This is a double challenge: The relevant documents must be found, and the documents retrieved must be relevant. The huge number of documents on the Web slightly modifies this problem. We conjecture that a web user will rarely read all the relevant documents. Thereby, retrieving the most relevant documents is more important than retrieving most of the relevant documents. According to this conjecture, we choose to evaluate and to compare our different models with the *precision* function which is not affected by the number of relevant documents that have been forgotten but that only measures the purity of the answer.

The choice between *macro averaging* and *micro averaging* is not fundamental for the Alleskl ar Dataset because the documents are quite equally distributed over the different categories. This is not true for WebKB as more than 45% of its documents are stored under the hold all category `other`. A micro averaging evaluation of the classifiers on WebKB emphasizes the models that correctly classify the most populated category, which means the hold on category of WebKB. Furthermore, micro averaging is often more enthousiastic than macro averaging because the most populated categories are better learned. That's why we evaluate our classifiers with a *macro averaging of the precision*.

6.1.2 Decision function based feature ranking

The linear support vector machines dispose the documents in an orthogonal vector space whose orthonormal base is formed by the features. After having stored all the documents of the training set, they determine the optimal hyperplane separating the positive and the negative examples. Classification is then made by looking at which side of the separation hyperplane the documents are disposed. This is done thanks to the decision function $D(\vec{x})$ (6.1):

$$D(\vec{x}) = sign(\vec{w} \cdot \vec{x} + b) \tag{6.1}$$

where $(\vec{w}, b)$ are the coordinates of the separation hyperplane and $\vec{w}$ its normal vector.

$$\vec{w} = \sum_{i=0}^{d-1} w_i \vec{j_i}$$

With $(\vec{j_i})_{i=0}^{d-1}$ orthonormal base of the vector space formed by the features and d dimension of the vector space (dimensionality of the classification problem).

The bigger the component w_i of the vector $\vec{w}$, the stronger the influence of feature i on the classification. The features with a big positive component promote a positive classification. Those whose component is next to zero are not a great influence on the classification and those with a big negative value promote a negative classification. This feature ranking technique is tested with success by Guyon in [9].

Combined with disjunct feature subsets representing different feature mining methods, this feature ranking lets us evaluate the relative information gain brought by each feature mining method: As the mining method feature subsets are disjunct, their corresponding vector subspaces are in direct sum and

$$E = M_1 \oplus M_2 \oplus \cdots \oplus M_n, \text{ with} \begin{cases} E & \text{the global vector space} \\ n & \text{number of mining methods} \\ M_i & \text{the subspaces of features mined by the } i^{th} \text{ method} \end{cases}$$

We decompose the normal vector $\vec{w}$ in $(\vec{w_i})_{i=1}^{n}$, with $\vec{w} = \sum_{i=1}^{n} \vec{w_i}$, $\forall i \in [1, n], \vec{w_i} \in M_i$ and rank the mining methods with two efficiency estimators:

feature estimator

$$e_f(m) = \frac{e_g(m)}{|M|}$$

The *feature estimator* measures the average information brought by one feature mined by the method m

mining method estimator

$$e_g(m) = |\vec{w_m}| = \sqrt{\sum_{f \in M} w_f^2}$$

The *mining method estimator* measures the information brought by all the features mined by the method m.

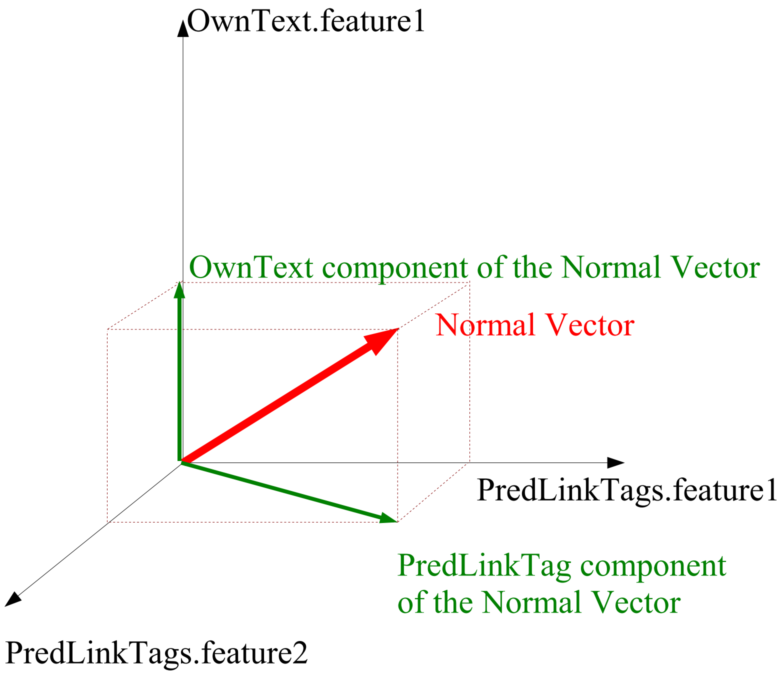

Figure 6.1: Decomposition of the normal vector

6.2 The sources of features

We compare in this section the sources of features with the help of the decision function based feature ranking, we make the definition of neighborhood of an anchor more precise. Then we present classification results using a single source of features and using any combination of two sources of features.

6.2.1 Comparison between the features

We make a document based feature ranking for each dataset (tables 6.1, 6.2, 6.3). These rankings show that even if OwnText carries a high number of features, each of these features contain very few information compared to those extracted by PredlistHeadings which are fewer but more informative. This experiment confirms that the anchor tags (PredLinkTags) are a very good source of information for classifying the targets and it confirms the results of Chakrabarti who shows in [5] that using only the content of the predecessors (PredText) can increase the error rate. We consequently decided not to use the whole text of the predecessors as feature for our classifiers. The average feature gain we obtain for WordsAround is much lower than PredLinktags because we defined here the neighborhood of a link as the 30 words before the link and the 30 words after the link, which is very large and collects a high number of spurious words mined without significantly increasing the information gain. Thereby, it is necessary to determine how wide the neighborhood's scope should be.

Feature	*Number of different words extracted*
PredLinkParagraph	79588
WordsAround	41513
OwnText	37898
PredHeadings	32832
PredLinkTags	4211
PredListHeadings	4118

Table 6.1: Ranking of the features mining for Allesklar

Feature	*Method component length*
PredLinkParagraph	51831
WordsAround	14360
PredHeadings	13070
OwnText	12658
PredListHeadings	4319
PredLinkTags	2594

Table 6.2: Ranking of the features mined for Alleksklar

6.2.2 Neighborhood of an anchor

Contrarily to the anchor description, the notion of neighboring words is vague and has to be made more precise. We computed the *macro precision* for each possible combination of 0 to 30 words before the anchor and 0 to 30 words after the anchor (figures 6.2 and 6.3).

Feature	*average feature length (method component length/features count)*
PredListHeadings	1.05
PredLinkParagraph	0.65
PredLinkTags	0.62
PredHeadings	0.40
AordsAround	0.35
OwnText	0.33

Table 6.3: Ranking of the average importance of a feature (Allesklar)

This experiment shows that the precision evolves similarly with words mined before the link and with words mined after the link. The determining criterion is not the position of the words taken in the neighborhood but their number.

In other words, the function of two variables

$$precision(Before, After)$$

can be approximated with a good accuracy by the function of one variable

$$precision(Words), \text{ with } Words = After + Before$$

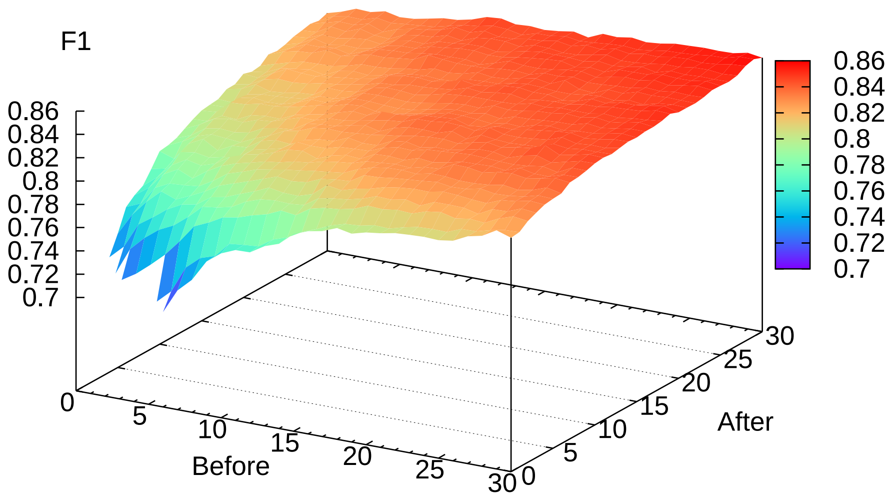

Figure 6.2: Macro precision of Allesklar for WordsAround for different values of before and after

The hyperlink graph of WebKB is too weakly connected to get significant results. But the good connectivity of Alleskar lets us verify this rule: Figure 6.4 is divided into two different parts: Before 20 words, the precision increases quickly. After 20 words, the precision still increases but very slowly while the dimensionality (the complexity of the classification problem) still grows. The best compromise for the scope of the neighborhood is thus 20, which we distribute equally before and after the anchor (10 words before the anchor and 10 words after).

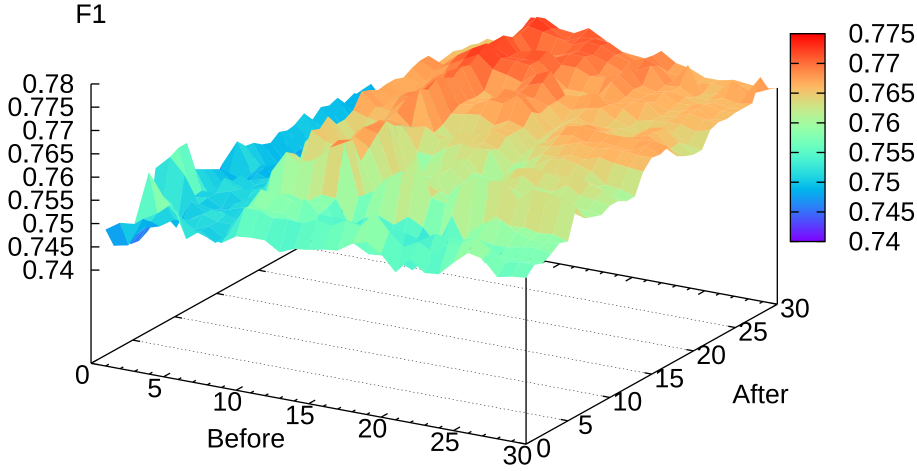

Figure 6.3: Macro precision of WebKB for WordsAround for different values of before and after

6.2.3 Using one feature

For this experiment, we store the non-local features in a *meta predecessor*, and use the *One against all* binarization. The following settings are common to this experiment and to the following ones: The size of the neighborhood has been fixed to 20 words: 10 words before the anchor and 10 words after the anchor, the text of the anchor is excluded. The macro precision is computed through a ten-folds cross validation for Alleskar. The WebKB dataset is a collection of web pages coming from five different universities. The five folds cross-validation implemented for this dataset separates the different universities, trains the classifiers on four universities and tests them on the fifth one.

We summarize the results in table 6.4. In each cell, the two first lines represent the precision and the recall reached and the third line is dedicated to the number of documents in the dataset that were covered by the feature pattern.

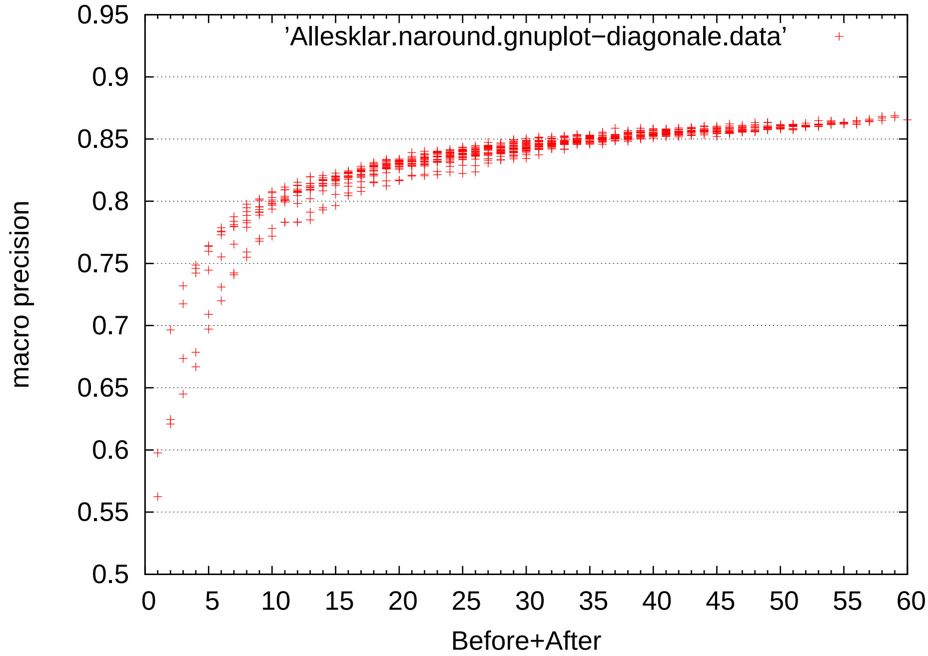

Figure 6.4: $precision(Before + After)$ of Alleskl ar for WordsAround

On both sets, the pattern that covers the most examples is OwnText. The good connectivity of Alleskl ar is translated here by an almost as good coverage for the non-local rules WordsAround and PredLinkTags. The slight difference between the coverages of PredLinkTags and WordsAround shows that not all anchor tags own a description. Therefore, looking at their neighborhood brings informations for links that could not be classified using only the anchor description. Structural headings and paragraphs of the links occur a bit less often and the most rare pattern is PredListHeadings, which is understood easily because this feature can be mined only when both following conditions are verified: The link is member of an HTML list and there are headings preceeding this list. However, fast half of the documents of Alleskl ar own at least one predecessor that satisfies this double condition. On the contrary, the weak connectivity of WebKB makes its non-local features mined only one-third as often as the local pattern OwnText.

As already shown by numerous studies, the anchor description or PredLinkTags pattern may outperform the local features. But the by far best precision reached for Alleskl ar is given by the neighborhood of the links. Alone, it outperforms traditional text classification by more than 43%. The precisions reached by the non-local features of WebKB are all lower than the precision of OwnText. However, we will show later that their diversity and their redundance allow combinations of non-local features that outperform traditionnal text classification by far.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Alleskla r	π=84.65% ρ=67.3% 3664	π=80% ρ=43.48% 3653	π=70.18% ρ=26.66% 1870	π=71.8% ρ=29.33% 2672	π=79.15% ρ=34.3% 2715	π=71.67% ρ=32.17% 3831
WebKB	π=41.07% ρ=17.94% 3007	π=35.54% ρ=21.35% 3653	π=17.38% ρ=14.89% 1644	π=28.35% ρ=17.37% 2828	π=29.17% ρ=16.71% 1144	π=45.37% ρ=24.71% 8277

Table 6.4: precision, recall and coverage reached using a single feature pattern on Alleskl ar and on WebKB

6.2.4 Combining two sources of features

Combining different sources of features affects the classification by several antagonist manners. On the one hand, it increases the amount of information collected about the examples and thereby helps the classification. On the other hand, it increases the dimensionality of the classification problem and it increases the coverage of the features mining and thus the diversity of the training set, which makes the training phase more complex.

In tables 6.5 and 6.6, we summarize the results of the classification experiments using a *meta predecessor*, the *one against all* binarization and any possible pair of feature sources. Each cell shows on the first lines the macro-precision (π) and the macro recall (μ) of the classification whereby the two feature sources shown in abscissa and ordinate are used together. The third line corresponds to the number of documents of the dataset that are covered by the feature mining rules.

The results of the light gray diagonal of the table are the ones that are obtained with only one feature rule. As the combination of two feature sources is commutative, each result appears twice in the table. For each of these pairs of cells, one has a white background and the other one is darkened. We write **Fett** the combinations that outperform each one of the source patterns alone.

Using two patterns instead of one does not always increase the precision. So that the positive effects of the combination prevail on the negative ones, the precision of the two patterns must be near. If there is a too large difference between the two precisions, the features brought by the least performing pattern are upset. They don't ameliorate much the classification performances. On the contrary, they increase the dimensionality of the problem and therefore hinder the training. There is however a case where combining two patterns increases the precision even if the single precisions' difference is significant. That is when the patterns target distinct features of the documents. For example the combination between the local pattern OwnText and any non-local pattern improves the precision for Alleskla r, and the Headings of a links list are too far from the anchor so that their words are mined by the pattern WordsAround.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=84.65% ρ=67.3% 3664	**π=84.89% ρ=65.67%** 3678	**π=84.87% ρ=67.31%** 3665	π=84.15% ρ=63.8% 3665	π=82.72% ρ=58.88% 3667	π=82.58% ρ=58.44% 3898
Pred LinkTags	**π=84.89% ρ=65.67%** 3678	π=80% ρ=43.48% 3653	**π=80.01% ρ=42.15%** 3653	π=76.68% ρ=38.5% 3653	π=76.44% ρ=36.19% 3655	π=75.75% ρ=37.1% 3898
PredList Headings	**π=84.87% ρ=67.31%** 3665	**π=80.01% ρ=42.15%** 3653	π=70.18% ρ=26.66% 1870	**π=71.83% ρ=28.78%** 2744	**π=79.66% ρ=26.77%** 3013	**π=72.36% ρ=33.82%** 3864
Pred Headings	π=84.15% ρ=63.8% 3665	π=76.68% ρ=38.5% 3653	**π=71.83% ρ=28.78%** 2744	π=71.8% ρ=29.33% 2672	π=70.09% ρ=26.62% 3103	**π=72.34% ρ=35.11%** 3879
PredLink Paragraph	π=82.72% ρ=58.88% 3667	π=76.44% ρ=36.19% 3655	**π=79.66% ρ=26.77%** 3013	π=70.09% ρ=26.62% 3103	π=79.15% ρ=34.3% 2715	π=72.51% ρ=34.87% 3882
Own Text	π=82.58% ρ=58.44% 3898	π=75.75% ρ=37.1% 3898	**π=72.36% ρ=33.82%** 3864	**π=72.34% ρ=35.11%** 3879	π=72.51% ρ=34.87% 3882	π=71.67% ρ=32.17% 3831

Table 6.5: Macro precision using two features on Allesklar

6.3 Ranking of the different methods

In this section, we study the influence of the choice between *Meta predecessor* and *Hyperlink Ensembles*, between the binarization algorithms *One against all* or *Round Robin* and between the mutualizations *Merging* or *Tagging*.

We test different methods to solve the multi-class problem of classification examples with non-local data mined with various patterns. We compare the 12 possible assemblages associating a combination process of the feature patterns, a binarization method and an algorithm for uniting the features of the different predecessors. In a first experiment, we run the classification process for the 6 feature sources available (Words Around, PredLinkTags, PredlistHeadings, PredHeadings, PredLinkParagraph and OwnText) and for the 15 combinations of two of those features. We rank the 12 assemblages for each of those 21 atomic experiments and give 12 points to the best method, 11 to the second, ... and finally 1 point to the least performing assemblage. The points gained with the atomic experiments are summed to obtain the general ranking shown in tables 6.7 and 6.8.

The results are very readable for the Allesklar Dataset. They lead us to the conclusion that whatever the combination process and the uniting algorithm are, the binarization

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=41.07% ρ=17.94% 3006	**π=56.66%** **ρ=20.35%** 3016	π=30.13% ρ=16.91% 3007	π=36.49% ρ=17.36% 3016	π=35.51% ρ=19.08% 3011	π=44.27% ρ=24.31% 8276
Pred LinkTags	**π=56.66%** **ρ=20.35%** 3016	π=35.54% ρ=21.35% 2940	π=34.02% ρ=19% 2941	π=29.23% ρ=17.36% 3001	π=30.53% ρ=19.87% 2954	π=43.23% ρ=24.44% 8276
PredList Headings	π=30.13% ρ=16.91% 3007	π=34.02% ρ=19% 2941	π=17.38% ρ=14.89% 1644	π=27.86% ρ=17.3% 2832	π=26.14% ρ=16.65% 2402	π=43.71% ρ=24.02% 8276
Pred Headings	π=36.49% ρ=17.36% 3016	π=29.23% ρ=17.36% 3001	π=27.86% ρ=17.3% 2832	π=28.35% ρ=17.37% 2828	π=26.13% ρ=16.84% 2911	π=43.96% ρ=23.65% 8276
PredLink Paragraph	π=35.51% ρ=19.08% 3011	π=30.53% ρ=19.87% 2954	π=26.14% ρ=16.65% 2402	π=26.13% ρ=16.84% 2911	π=29.17% ρ=16.71% 1143	π=43.5% ρ=24.69% 8276
Own Text	π=44.27% ρ=24.31% 8276	π=43.23% ρ=24.44% 8276	π=43.71% ρ=24.02% 8276	π=43.96% ρ=23.65% 8276	π=43.5% ρ=24.69% 8276	π=45.37% ρ=24.71% 8276

Table 6.6: Macro precision using two features on WebKB

One against all outperforms *Round Robin* by about 35%. However, this gain of precision is paid with an important loss of recall. We explain this phenomenon in the subsection 6.3.2. Whatever the other solutions have been adopted to solve the two other problems, the uniting algorithm *Meta Predecessor* outperforms *Hyperlink Ensembles* by about 10% while *Hyperlink Ensembles* outperforms *Meta learned Hyperlink Ensembles* by about 6%. Finally, the combination process *Merging* outperforms *Tagging* by about 2%.

Unfortunately, the analysis of the results for the WebKB dataset is not as obvious as Allesklar's. While the points distribution induces a clear ranking for Allesklar where the best method receives seven times as much points as the last one, this separation is not as clear for WebKB where the first method only receives twice as many point as the last one. This results in a bigger proximity in the values of precision and recall. However, the ranking that has been be lead shows that *One against all* outperforms *Round Robin*. Nevertheless, in this case, *Meta learned Hyperlink Ensembles* outperform *Hyperlink Ensembles* while *Meta Predecessor* outperform both of them. Furthermore, *Tagging* outperforms *Merging*. We explain this phenomenon in section 6.3.3.

Combination	**Binarization**	**non-local**	**points**	**average π**	**average ρ**
Merging	One against all	Meta predecessor	240	78.37%	43.76%
Tagging	One against all	Meta predecessor	225	77.35%	42.25%
Merging	One against all	Hyperlink Ensembles	204	73.17%	33.42%
Tagging	One against all	Hyperlink Ensembles	193	72.43%	32.51%
Merging	One against all	Meta learning	147	68.98%	37.77%
Tagging	One against all	Meta learning	139	67.85%	36.61%
Merging	Round Robin	Meta predecessor	129	66.32%	59.51%
Tagging	Round Robin	Meta predecessor	117	64.95%	57.95%
Merging	Round Robin	Hyperlink Ensembles	93	61.64%	48.36%
Tagging	Round Robin	Hyperlink Ensembles	73	59.83%	47.5%
Merging	Round Robin	Meta learning	42	57.71%	50.44%
Tagging	Round Robin	Meta learning	36	56%	48.62%

Table 6.7: Ranking of the different methods for Alleskiar

6.3.1 Meta Predecessor, Hyperlink Ensembles and Meta learned Hyperlink Ensembles

Meta Predecessor and Hyperlink Ensembles

See Result tables B.1, B.2, B.3, B.4 for Alleskiar and B.5, B.6, B.7, B.8 for WebKB

Our experiments show that employment of Hyperlink Ensembles must be combined with careful precautions. The key principle of Hyperlink Ensembles is to discard the features coming from the noisy predecessors by choosing the majority prediction between the predecessors. One required condition is thus that the predecessors on which prediction helpful information is mined are correctly classified. But by splitting the classification problem of one page owning n predecessors into n classification problems, we divide by n the number of features representing each example while the dimensionality of the learning task is kept. For example, a page of the category *Work and Jobs* has a high probability to own at least one predecessor containing the word `employment`. The words `colleagues`, `coffee` or `boss` appear more rarely. A Meta Predecessor learner just has to keep the rule

$$\text{employment} \longrightarrow \textit{Work and Jobs}$$

But a Hyperlink Ensemble classifier must correctly behave when the word `coffee` appears and not `employment`, `boss` and `colleagues`, or when the word `colleagues` appears and not `employment`, `boss` and `coffee`. It must thus keep the rules

$$\begin{aligned} \text{employment} &\longrightarrow \textit{Work and Jobs} \\ \text{colleagues} &\longrightarrow \textit{Work and Jobs} \\ \text{boss} &\longrightarrow \textit{Work and Jobs} \\ \text{coffee} &\longrightarrow \textit{Work and Jobs} \end{aligned}$$

Combination	**Binarization**	**non-local**	**points**	**average π**	**average ρ**
Tagging	One against all	Meta predecessor	174	35.63%	19.75%
Tagging	One against all	Hyperlink Ensembles	169	33.74%	18.9%
Merging	One against all	Meta predecessor	160	34.5%	19.22%
Tagging	One against all	Meta learning	158	32.51%	19.25%
Tagging	Round Robin	Meta predecessor	145	35.15%	21.65%
Merging	Round Robin	Meta predecessor	144	34.51%	21.06%
Tagging	Round Robin	Meta learning	140	33.15%	21.33%
Tagging	Round Robin	Hyperlink Ensembles	136	32.5%	20.59%
Merging	Round Robin	Meta learning	107	29.98%	21.68%
Merging	Round Robin	Hyperlink Ensembles	106	28.11%	18.3%
Merging	One against all	Meta learning	104	28.41%	19.52%
Merging	One against all	Hyperlink Ensembles	95	27.21%	16.97%

Table 6.8: Ranking of the different methods for WebKB

In other words, the number of clusters in the vector space representation of the dataset is higher with Hyperlink Ensembles than with a Meta Predecessor. Hyperlink Ensembles increase the VC-dimension of the classification problem.

As a consequence, employing *Hyperlink Ensembles* shall be restricted to cases where the dimensionality of the problem is small, that is with mining methods gathering very pure and accurate features. *Hyperlink Ensembles* shall be limited to cases where the amount of features collected for each predecessor is sufficient to have the classification be relevant. More generally, *Hyperlink Ensembles* is a powerful method to discard spurious predecessors if it relies on a powerful classifier. *Hyperlink Ensembles* do not help in the case of homogeneous and related predecessors that are more or less correctly classified by a low confidence classifier. Under those conditions, our experiments show that *Hyperlink Ensembles* outperform *Meta Predecessor* for WebKB in most of the combinations between PredHeadings, PredListHeadings and PredLinkParagraph and in some of those combinations for Alleskla r.

Meta learned Hyperlink Ensembles

See *Result tables B.9, B.10, B.11, B.12 for Alleskla r and B.13, B.14, B.15, B.16 for WebKB*

In order to circumvent the problem of dimensionality growth with *Hyperlink Ensembles*, we test a mix solution consisting of using *Meta Predecessors* on the training set for the learning phase, and to use the models obtained on *Hyperlink Ensembles*. Whereas the type of objects on which the classifiers are trained and on which they are used differ, this method sometimes outperform *Hyperlink Ensembles* with a big precision gap. However, this method never outperforms *Meta Predecessor*.

6.3.2 One against all and Round Robin

See *Result tables B.17, B.18, B.19, B.20 for Alleskl ar and B.21, B.22, B.23, B.24 for WebKB*

On both of the datasets, *One against all* outperforms *Round Robin* in a comfortable majority of experiments. Those results shall however be precisely analysed. Each *One against all* category-specific classifier is asked to decide between a very strait class, the positive one, and a much wider one which is the aggregation of all the other categories. In many cases, the category-specific classifier chooses the widest class which is the negative one.

	as 1	**as 2**	**as 3**	**as 4**	**as 5**	ρ	F_1
is 1	802	8	4	3	1	0.98	0.454
is 2	531	248	1	8	4	0.313	0.467
is 3	580	4	154	7	1	0.206	0.338
is 4	391	2	1	345	2	0.465	0.609
is 5	399	7	3	27	120	0.215	0.349
π	0.296	0.921	0.944	0.884	0.937		

Table 6.9: Confusion Matrix for *Alleskl ar* using *One against all*, combination PredLinkTags and PredListHeadings, with Merging and Meta Predecessor

	as 1	**as 2**	**as 3**	**as 4**	**as 5**	**as 6**	**as 7**	ρ	F_1
is 1	2142	29	10	10	8			0.974	0.848
is 2	182	17	1		1			0.084	0.136
is 3	195							0	0
is 4	155		6	12				0.069	0.121
is 5	116							0	0
is 6	38							0	0
is 7	18			1				0	0
π	0.752	0.369	0	0.521	0				

Table 6.10: Confusion Matrix for *WebKB* using *One against all*, combination PredLinkTags and PredListHeadings, with Merging and Meta Predecessor

For numerous pages of the dataset, all the category-specific classifiers say *no* and all the categories receive the same amount of points: $n - 1$. Thereby, the end-predicted category is the most populated one. Many examples are thus classified under the biggest category. Much more than needed. On the one hand, the precision of this category is low. But on the other hand, the examples that are classified under an other category are examples whose corresponding category-specific classifier said *yes* while the negative class was much wider. The probability that the classification is correct is thus high.

The *One against all* binarization sacrifies the precision of the most populated category and grants the other categories a high level of precision but a low recall. When the macro precision is computed, the low precision of the most populated category is attenuated by the $n-1$ good precision levels of the other categories.

	as 1	**as 2**	**as 3**	**as 4**	**as 5**	ρ	F_1
is 1	606	57	128	16	11	0.74	0.621
is 2	166	455	129	26	16	0.574	0.631
is 3	146	50	503	32	15	0.674	0.561
is 4	90	35	147	453	16	0.611	0.684
is 5	122	50	137	56	191	0.343	0.474
π	0.536	0.703	0.481	0.777	0.767		

Table 6.11: Confusion Matrix for *Allesklar* using *Round Robin*, combination PredLinkTags and PredListHeadings, with Merging and Meta Predecessor

	as 1	**as 2**	**as 3**	**as 4**	**as 5**	**as 6**	**as 7**	ρ	F_1
is 1	2180	16	22	9	14	1	2	0.971	0.85
is 2	169	31	2					0.153	0.248
is 3	182		15					0.076	0.125
is 4	168		1	4			3	0.022	0.04
is 5	118		1	1	1			0.008	0.014
is 6	42							0	0
is 7	19							0	0
π	0.757	0.659	0.365	0.285	0.066	0	0		

Table 6.12: Confusion Matrix for *WebKB* using *Round Robin*, combination PredLinkTags and PredListHeadings, with Merging and Meta Predecessor

With the *Round Robin* binarization, the binary classifiers cannot output a default prediction. They have no choice but to give one category their preference. Thereby, the conflicts do not involve all the categories but only the few ones that get the best score. As a consequence, the incorrectly classified examples are more equally distributed among the categories. The precision rates of all the categories are lowered by this more democratic distribution of the undecided pages which hinders the macro average precision more than with *One against all.*

An idea for improving the binarization could be to make two category predictions. The first one with *One against all* and the second one with *Round Robin.* The end prediction would be validated only if the two intermediate predictions agree. Otherwise, the example would be labeled as *Undefined* and thrown away.

6.3.3 Tagging and Merging

See *Result tables B.25, B.26, B.27, B.28 for Alleskl ar and B.29, B.30, B.31, B.32 for WebKB*

The experiments led on WebKB and Alleskl ar show that Merging is more accurate for Alleskl ar and that Tagging if often more accurate for WebKB. More precisely, we can identify groups of feature patterns that work better together when they are merged for WebKB: The *Headings group* composed of *PredHeadings* and *PredListHeadings*, the *Link group* composed of *PredLinkTags* and *WordsAround* and finally the *Text group* formed by *OwnText* and *PredLinkParagraph*.

As explained in paragraph 3.6, Merging keeps the information of redundancy but erases the origin of the features. Knowing that a word occurs three times among the various sources gives a clue that this word is important for the classification. But knowing that a given word has been mined on a very representative place like the heading and not in the crowd of the words around a link makes him a particularly interesting too for the classification. Both Merging and Tagging methods are thus not optimal because each one looses a part of the classification information.

The results of this experiment are not surprising because the weak connectivity of WebKB prevents the redundancy kept by merging to be significant: The average in-degree is too low so that common clue words can be mined on several predecessors. Thereby, *Tagging* erases on WebKB a weak redundancy while *Merging* looses the location information. On the contrary, the good connectivity of Alleskl ar favorises *Merging*.

We propose a framework for defining an optimal method bringing the features coming from different sources together loosing without the informations of redundancy or origin. Our proposal is to use a weighted merging that gives a greater importance to the pure and accurate feature sources than to the features coming from spurious sources. Instead of determining which one between Merging or Tagging looses less information, this method would aggregate their respective heavy points.

6.4 Best model

The preceding experiments show that the best results are obtained with the *One against All* binarization, with a *Meta Predecessor* and by tagging the features with their origin. The best features source appears to be the anchor description in the predecessors combined with the words neighboring this anchor. We show in this section detailed results for this best model for Alleskl ar and for WebKB.

6.4.1 Alleskl ar

We present here the confusion matrix for the classification of Alleskl ar:

	as 1	as 2	as 3	as 4	as 5	ρ	F1
is 1	794	11	10	6	3	0.963	0.575
is 2	332	444	8	7	2	0.559	0.706
is 3	196	1	552	1	3	0.733	0.823
is 4	352	1	5	390	1	0.52	0.675
is 5	258	5	12	1	283	0.506	0.664
π	0.41	0.961	0.94	0.962	0.969		

and the efficiency measures

micro accuracy	0.868
micro error	0.132
micro precision	0.670
micro recall	0.670
micro F_1	0.670
macro accuracy	0.868
macro error	0.132
macro precision	0.849
macro recall	0.657
macro F_1	0.690

The text-only classifier's macro precision is 71.67% on this dataset in the same conditions. Our model outperforms the traditional text classifier by nearly 18.5%. As we used the One Against All binarization, the first category's precision is however low. A model which would detect which examples are classified by default and which would throw them out instead of trying to find the least bad category would have a better macro precision (but of course a lower recall).

6.4.2 WebKB

We present here the confusion matrix for the classification of WebKB:

	as 1	as 2	as 3	as 4	as 5	as 6	as 7	ρ	F1
is 1	2253	4	1		2			0.996	0.868
is 2	132	70						0.346	0.506
is 3	186		11					0.055	0.103
is 4	173			3				0.017	0.033
is 5	120				1			0.008	0.015
is 6	41							0	0
is 7	19							0	0
π	0.77	0.945	0.916	1	0.333	0	0		

and the efficiency measures

micro accuracy	0.936
micro error	0.064
micro precision	0.775
micro recall	0.775
micro F_1	0.775
macro accuracy	0.936
macro error	0.064
macro precision	0.567
macro recall	0.204
macro F_1	0.219

The text-only classifier's macro precision is 45.37% on this dataset in the same conditions. Our model outperforms the traditional text classifier by more than 24.8%. However, the cost for this gain of precision if a heavy reduction of the coverage. Only 3016 examples can be classified by the best classifier while there are more than 8000 examples in the whole dataset.

Chapter 7

Conclusion

In this diploma thesis, various methods for using both local and non-local features have been investigated to classifying Web pages. The biggest advantage of our model is to use concurrently the HTML intern structure of the web pages and the Hyperlink graph structure of the Web. We mined features on various strategic locations of the web page and of its predecessors and we generate a global end prediction.

Four main problems arose while we conceived our model:

- Most of the classification algorithms only can express a prediction between two classes while we had to decide between several categories. We tested two binarization algorithms to solve this problem, namely *One against All* and *Round Robin*.
- We should decide wether the non-local features shall be computed separately in *Hyperlink Ensembles* or shall be brought together in a *Meta Predecessor* before working on them.
- We should determine how two features mined on two different strategic locations should be put together in order to have them help together the classification. The first solution studied was to merge them like features coming from a unique localization. The second one was to consider a same word mined on two different localizations like two different words.
- The last problem was finally to find which feature locations are helpful for the classification.

We implemented all those solutions, evaluated them and compared them on two Datasets. The first one named *Alleskiar* has been collected specifically for this study on a German Web directory and the second one named *WebKB* had already been tested in various studies.

Our model outperforms a traditional text classifier by up to 25% but we do not only validate our model. We present ideas that motivate further work for improving it, especially for taking full advantage of the Hyperlink Ensembles and of the Round Robin binarization. Those research trails are

- Insert a meta-learner that reads the predictions of a *Round Robin* classifier and a *One against All* classifier and that computes a global prediction.
- Develop a solution which aggregates the heavy points of *Merging* and *Tagging*
- Study how the different feature sources can be brought together for improving concurrently the precision and the coverage of the hyperlink-based classifier.

Bibliography

[1] Xpath tutorial. http://www.zvon.org/xxl/XMLTutorial/General/book.html.

[2] Erin L. Allwein, Robert E. Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. In *Proceedings of the 17th International Conference on Machine Learning*, pages 9–16. Morgan Kaufmann, San Francisco, CA, 2000.

[3] U. Borghoff, H. Karch, and J. Schlichter. Constraint-based information gathering for a network publication system. In *Practical Applications of Agent Methodology 1996, London, United Kingdom*, page 45, 1996.

[4] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7):107–117, 1998.

[5] Soumen Chakrabarti, Byron Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. In Laura M. Haas and Ashutosh Tiwary, editors, *SIGMOD 1998, Proceedings Association for Computing Machinery Special Interest Group on Management Of Data, International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA*, pages 307–318. ACM Press, 1998.

[6] Thomas G. Dietterich. Ensemble methods in machine learning. *Lecture Notes in Computer Science*, 1857:1–15, 2000.

[7] Susan Dumais, John Platt, David Heckerman, and Mehran Sahami. Inductive learning algorithms and representations for text categorization. In *CIKM '98: Proceedings of the seventh international conference on Information and knowledge management*, pages 148–155, New York, NY, USA, 1998. ACM Press.

[8] Johannes Fürnkranz. Hyperlink ensembles: A case study in hypertext classification.

[9] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, 2002.

[10] Min-Yen Kan. Web page classification without the web page. In *WWW Alt. '04: Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, pages 262–263, New York, NY, USA, 2004. ACM Press.

[11] Steve Lawrence and C. Lee Giles. Searching the world wide web. *Science*, 280:98–100, 1998.

[12] Qing Lu and Lise Getoor. Link-based classification. In *International Conference on Machine Learning*, pages 496–503, 2003.

[13] Robert E. Schapire, Yoram Singer, and Amit Singhal. Boosting and Rocchio applied to text filtering. In W. Bruce Croft, Alistair Moffat, Cornelis J. van Rijsbergen, Ross Wilkinson, and Justin Zobel, editors, *Proceedings of Special Interest Group on Information Retrieval SIGIR-98, 21st ACM International Conference on Research and Development in Information Retrieval*, pages 215–223, Melbourne, AU, 1998. ACM Press, New York, US.

[14] Hinrich Schutze, David A. Hull, and Jan O. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Research and Development in Information Retrieval*, pages 229–237, 1995.

[15] Michael F. Schwartz, Alan Emtage, Brewster Kahle, and B. Clifford Neuman. A comparison of internet resource discovery approaches. *Computing Systems*, 5(4):461–493, 1992.

[16] Fabrizio Sebastiani. Machine learning in automated text categorization. *Association for Computing Machinery Computing Surveys*, 34(1):1–47, 2002.

[17] Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *Special Interest Group on Information Retrieval SIGIR '99: Proceedings of the 22nd annual international ACM Special Interest Group on Information Retrieval SIGIR conference on Research and development in information retrieval*, pages 42–49, New York, NY, USA, 1999. ACM Press.

[18] Xiao-Feng Zhang, Chak-Man Lam, and William K. Cheung. Web page organization and visualization using generative topographic mapping - a pilot study. 2004.

Appendix A

Stop Words List

Those stopwords have been downloaded on http://www.ranks.nl/stopwords/

A.1 Common stop words

D _

A.2 German stop words

aber
als
am
an
auch
auf
aus
bei
bin
bis
bist
da
dadurch
daher
darum
das
daß
dass
dein
deine
dem
den
der
des
dessen
deshalb
die
dies
dieser
dieses
doch
dort
du
durch
ein
eine
einem
einen
einer
eines
er
es
euer
eure
für
hatte
hatten
hattest
hattet
hier
hinter
ich
ihr
ihre
im
in
ist
ja
jede
jedem
jeden
jeder
jedes
jener
jenes
jetzt
kann
kannst
koennen
koennt
machen
mein
meine
mit
muss
musst
musst
muessen
muesst
nach
nachdem
nein
nicht
nun
oder
seid
sein
seine
sich
sie
sind
soll
sollen
sollst
sollt
sonst
soweit
sowie
und
unser
unsere
unter
vom
von
vor

wann	wenn	weshalb	wird	zu
warum	wer	wie	wirst	zum
was	werde	wieder	wo	zur
weiter	werden	wieso	woher	ueber
weitere	werdet	wir	wohin	

A.3 English stop words

a	back	detail	forty	inc
about	be	do	found	indeed
above	became	done	four	interest
across	because	down	from	into
after	become	due	front	is
afterwards	becomes	during	full	it
again	becoming	each	further	its
against	been	eg	get	itself
all	before	eight	give	keep
almost	beforehand	either	go	last
alone	behind	eleven	had	latter
along	being	else	has	latterly
already	below	elsewhere	hasnt	least
also	beside	empty	have	less
although	besides	enough	he	ltd
always	between	etc	hence	made
am	beyond	even	her	many
among	bill	ever	here	may
amongst	both	every	hereafter	me
amoungst	bottom	everyone	hereby	meanwhile
amount	but	everything	herein	might
an	by	everywhere	hereupon	mill
and	call	except	hers	mine
another	can	few	herself	more
any	cannot	fifteen	him	moreover
anyhow	cant	fify	himself	most
anyone	co	fill	his	mostly
anything	computer	find	how	move
anyway	con	fire	however	much
anywhere	could	first	hundred	must
are	couldnt	five	i	my
around	cry	for	ie	myself
as	de	former	if	name
at	describe	formerly	in	namely

```
neither
never
nevertheless
next
nine
no
nobody
none
noone
nor
not
nothing
now
nowhere
of
off
often
on
once
one
only
onto
or
other
others
otherwise
our
ours
ourselves
out
over
own
part
per
perhaps
please
put
rather
re
same
see
seem
seemed
seeming
seems
serious
several
she
should
show
side
since
sincere
six
sixty
so
some
somehow
someone
something
sometime
sometimes
somewhere
still
such
system
take
ten
than
that
the
their
them
themselves
then
thence
there
thereafter
thereby
therefore
therein
thereupon
these
they
thick
thin
third
this
those
though
three
through
throughout
thru
thus
to
together
too
top
toward
towards
twelve
twenty
two
un
under
until
up
upon
us
very
via
was
we
well
were
what
whatever
when
whence
whenever
where
whereafter
whereas
whereby
wherein
whereupon
wherever
whether
which
while
whither
who
whoever
whole
whom
whose
why
will
with
within
without
would
yet
you
your
yours
yourself
yourselves
```

Appendix B

Result Tables

B.1 Meta Predecessor and Hyperlink Ensembles

Green if for Meta Predecessor and blue for Hyperlink Ensembles

B.1.1 Alleskla r

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=84.65%** **ρ=67.3%** π=82.97% ρ=46.92% 3664	**π=84.89%** **ρ=65.67%** π=83.41% ρ=47.31% 3678	**π=84.87%** **ρ=67.31%** π=83.06% ρ=47.07% 3665	**π=84.15%** **ρ=63.8%** π=82.89% ρ=38.34% 3665	π=82.72% ρ=58.88% **π=83.25%** **ρ=45.75%** 3667	**π=82.58%** **ρ=58.44%** π=81.88% ρ=39.39% 3898
Pred LinkTags	**π=84.89%** **ρ=65.67%** π=83.41% ρ=47.31% 3678	**π=80%** **ρ=43.48%** π=75.82% ρ=37.58% 3653	**π=80.01%** **ρ=42.15%** π=76.67% ρ=35.77% 3653	**π=76.68%** **ρ=38.5%** π=72.03% ρ=29.86% 3653	**π=76.44%** **ρ=36.19%** π=75.75% ρ=31.4% 3655	**π=75.75%** **ρ=37.1%** π=73.92% ρ=30.45% 3898
PredList Headings	**π=84.87%** **ρ=67.31%** π=83.06% ρ=47.07% 3665	**π=80.01%** **ρ=42.15%** π=76.67% ρ=35.77% 3653	**π=70.18%** **ρ=26.66%** π=68.19% ρ=28.54% 1870	**π=71.83%** **ρ=28.78%** π=67.81% ρ=28.37% 2744	**π=79.66%** **ρ=26.77%** π=77.62% ρ=27.61% 3013	**π=72.36%** **ρ=33.82%** π=71.62% ρ=27.81% 3864
Pred Headings	**π=84.15%** **ρ=63.8%** π=82.89% ρ=38.34% 3665	**π=76.68%** **ρ=38.5%** π=72.03% ρ=29.86% 3653	**π=71.83%** **ρ=28.78%** π=67.81% ρ=28.37% 2744	**π=71.8%** **ρ=29.33%** π=66.41% ρ=29.12% 2672	π=70.09% ρ=26.62% **π=70.52%** **ρ=26.91%** 3103	π=72.34% ρ=35.11% **π=77.91%** **ρ=25.34%** 3879
PredLink Paragraph	π=82.72% ρ=58.88% **π=83.25%** **ρ=45.75%** 3667	**π=76.44%** **ρ=36.19%** π=75.75% ρ=31.4% 3655	**π=79.66%** **ρ=26.77%** π=77.62% ρ=27.61% 3013	π=70.09% ρ=26.62% **π=70.52%** **ρ=26.91%** 3103	**π=79.15%** **ρ=34.3%** π=74.94% ρ=30.62% 2715	π=72.51% ρ=34.87% **π=74.32%** **ρ=28.61%** 3882
Own Text	**π=82.58%** **ρ=58.44%** π=81.88% ρ=39.39% 3898	**π=75.75%** **ρ=37.1%** π=73.92% ρ=30.45% 3898	**π=72.36%** **ρ=33.82%** π=71.62% ρ=27.81% 3864	π=72.34% ρ=35.11% **π=77.91%** **ρ=25.34%** 3879	π=72.51% ρ=34.87% **π=74.32%** **ρ=28.61%** 3882	**π=71.67%** **ρ=32.17%** π=- ρ=- 3831

Table B.1: Alleskla r Tagging One Against All Meta Predecessor -Alleskla r Tagging One Against All Hyperlink Ensembles

Meta Predecessor outperforms Hyperlink Ensembles in almost all the cases. Only 4 combinations see the Hyperlink Ensembles have a better precision but their lead is then small.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=84.65%** **ρ=67.3%** π=82.97% ρ=46.92% 3664	**π=84.82%** **ρ=65.67%** π=84.23% ρ=51.09% 3678	**π=85.05%** **ρ=68.28%** π=83.08% ρ=46.96% 3665	**π=84.15%** **ρ=64.57%** π=83.09% ρ=38.21% 3665	**π=83.5%** **ρ=62.15%** π=83.43% ρ=48.43% 3667	**π=83.53%** **ρ=63.99%** π=83.24% ρ=43.5% 3898
Pred LinkTags	**π=84.82%** **ρ=65.67%** π=84.23% ρ=51.09% 3678	**π=80%** **ρ=43.48%** π=75.82% ρ=37.58% 3653	**π=79.71%** **ρ=43.63%** π=77.05% ρ=35.79% 3653	**π=77.74%** **ρ=40.17%** π=76.87% ρ=31.86% 3653	**π=78.43%** **ρ=39.77%** π=77.71% ρ=35.36% 3655	**π=79.14%** **ρ=41.05%** π=76.46% ρ=33.01% 3898
PredList Headings	**π=85.05%** **ρ=68.28%** π=83.08% ρ=46.96% 3665	**π=79.71%** **ρ=43.63%** π=77.05% ρ=35.79% 3653	**π=70.18%** **ρ=26.66%** π=68.19% ρ=28.54% 1870	**π=74.94%** **ρ=29.4%** π=66.75% ρ=27.95% 2744	**π=80.3%** **ρ=27.96%** π=78.48% ρ=28.62% 3013	π=73.39% ρ=35.21% **π=75.07%** **ρ=26.79%** 3864
Pred Headings	**π=84.15%** **ρ=64.57%** π=83.09% ρ=38.21% 3665	**π=77.74%** **ρ=40.17%** π=76.87% ρ=31.86% 3653	**π=74.94%** **ρ=29.4%** π=66.75% ρ=27.95% 2744	**π=71.8%** **ρ=29.33%** π=66.41% ρ=29.12% 2672	**π=74.2%** **ρ=29.17%** π=71.88% ρ=25.87% 3103	**π=73.88%** **ρ=36.11%** π=72.64% ρ=25.14% 3879
PredLink Paragraph	**π=83.5%** **ρ=62.15%** π=83.43% ρ=48.43% 3667	**π=78.43%** **ρ=39.77%** π=77.71% ρ=35.36% 3655	**π=80.3%** **ρ=27.96%** π=78.48% ρ=28.62% 3013	**π=74.2%** **ρ=29.17%** π=71.88% ρ=25.87% 3103	**π=79.15%** **ρ=34.3%** π=74.94% ρ=30.62% 2715	π=75.58% ρ=38.68% **π=78.35%** **ρ=30.39%** 3882
Own Text	**π=83.53%** **ρ=63.99%** π=83.24% ρ=43.5% 3898	**π=79.14%** **ρ=41.05%** π=76.46% ρ=33.01% 3898	π=73.39% ρ=35.21% **π=75.07%** **ρ=26.79%** 3864	**π=73.88%** **ρ=36.11%** π=72.64% ρ=25.14% 3879	π=75.58% ρ=38.68% **π=78.35%** **ρ=30.39%** 3882	**π=71.67%** **ρ=32.17%** π=- ρ=- 3831

Table B.2: Alleskiar Merging One Against All Meta Predecessor -Alleskiar Merging One Against All Hyperlink Ensembles
Meta Predecessor outperforms Hyperlink Ensembles in almost all the cases. Only 2 combinations see Hyperlink Ensembles have a better precision, but it is with Owntext, features group that is not the most important for Hyperlink Ensembles because Hyperlink Ensembles consider the target page just like one more predecessor.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=81.84%** **ρ=79.67%** π=77.83% ρ=72.85% 3664	**π=81.55%** **ρ=79.4%** π=77.59% ρ=73.19% 3678	**π=81.52%** **ρ=79.61%** π=77.32% ρ=72.06% 3665	**π=80.04%** **ρ=77.95%** π=75.58% ρ=67.67% 3665	**π=77.36%** **ρ=73.24%** π=76.66% ρ=70.12% 3667	**π=77.68%** **ρ=75.15%** π=73.67% ρ=65.76% 3898
Pred LinkTags	**π=81.55%** **ρ=79.4%** π=77.59% ρ=73.19% 3678	**π=64.12%** **ρ=57.35%** π=59.15% ρ=48.9% 3653	**π=63.87%** **ρ=57.07%** π=56.62% ρ=47.74% 3653	**π=59.22%** **ρ=53.72%** π=56.91% ρ=40.13% 3653	**π=59.38%** **ρ=52.68%** π=55.31% ρ=44.83% 3655	**π=63.11%** **ρ=56.63%** π=55% ρ=44.37% 3898
PredList Headings	**π=81.52%** **ρ=79.61%** π=77.32% ρ=72.06% 3665	**π=63.87%** **ρ=57.07%** π=56.62% ρ=47.74% 3653	**π=48.34%** **ρ=39.18%** π=47.69% ρ=33.97% 1870	π=54.43% ρ=42.67% **π=56.71%** **ρ=37%** 2744	**π=57.78%** **ρ=42.88%** π=56.28% ρ=37.08% 3013	**π=60.43%** **ρ=54.03%** π=54.29% ρ=42.65% 3864
Pred Headings	**π=80.04%** **ρ=77.95%** π=75.58% ρ=67.67% 3665	**π=59.22%** **ρ=53.72%** π=56.91% ρ=40.13% 3653	π=54.43% ρ=42.67% **π=56.71%** **ρ=37%** 2744	π=55.42% ρ=44.09% **π=59.11%** **ρ=37.65%** 2672	π=54.6% ρ=40.5% **π=61.39%** **ρ=38.76%** 3103	**π=61%** **ρ=54.56%** π=56.34% ρ=38.08% 3879
PredLink Paragraph	**π=77.36%** **ρ=73.24%** π=76.66% ρ=70.12% 3667	**π=59.38%** **ρ=52.68%** π=55.31% ρ=44.83% 3655	**π=57.78%** **ρ=42.88%** π=56.28% ρ=37.08% 3013	π=54.6% ρ=40.5% **π=61.39%** **ρ=38.76%** 3103	**π=64.88%** **ρ=51.51%** π=63.3% ρ=41.72% 2715	**π=60.88%** **ρ=55.4%** π=59.7% ρ=42.89% 3882
Own Text	**π=77.68%** **ρ=75.15%** π=73.67% ρ=65.76% 3898	**π=63.11%** **ρ=56.63%** π=55% ρ=44.37% 3898	**π=60.43%** **ρ=54.03%** π=54.29% ρ=42.65% 3864	**π=61%** **ρ=54.56%** π=56.34% ρ=38.08% 3879	**π=60.88%** **ρ=55.4%** π=59.7% ρ=42.89% 3882	**π=56.47%** **ρ=49.71%** π=- ρ=- 3831

Table B.3: Alleskla̲r Tagging Round Robin Meta Predecessor -Alleskla̲r Tagging Round Robin Hyperlink Ensembles
Meta Predecessor outperforms Hyperlink Ensembles in almost all the cases. Only 3 combinations see Hyperlink Ensembles have a better precision, all with *PredHeadings*.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=81.84%** **ρ=79.67%** π=77.83% ρ=72.85% 3664	**π=81.94%** **ρ=79.62%** π=78.87% ρ=74.84% 3678	**π=81.4%** **ρ=79.5%** π=77.76% ρ=72.58% 3665	**π=79.76%** **ρ=77.84%** π=76.14% ρ=67.9% 3665	**π=79.25%** **ρ=76.33%** π=78.15% ρ=72.95% 3667	**π=79.58%** **ρ=77.93%** π=78.69% ρ=73.23% 3898
Pred LinkTags	**π=81.94%** **ρ=79.62%** π=78.87% ρ=74.84% 3678	**π=64.12%** **ρ=57.35%** π=59.15% ρ=48.9% 3653	**π=65.31%** **ρ=58.89%** π=57.45% ρ=48.66% 3653	**π=61.02%** **ρ=55.21%** π=57.72% ρ=41.71% 3653	**π=64.68%** **ρ=56.27%** π=61.89% ρ=52.07% 3655	**π=67.81%** **ρ=60.71%** π=59.24% ρ=48.26% 3898
PredList Headings	**π=81.4%** **ρ=79.5%** π=77.76% ρ=72.58% 3665	**π=65.31%** **ρ=58.89%** π=57.45% ρ=48.66% 3653	**π=48.34%** **ρ=39.18%** π=47.69% ρ=33.97% 1870	π=55.1% ρ=43.64% **π=57.59%** **ρ=37.26%** 2744	**π=60.3%** **ρ=44.85%** π=56.74% ρ=38.36% 3013	**π=60.5%** **ρ=54.57%** π=59.29% ρ=34.18% 3864
Pred Headings	**π=79.76%** **ρ=77.84%** π=76.14% ρ=67.9% 3665	**π=61.02%** **ρ=55.21%** π=57.72% ρ=41.71% 3653	π=55.1% ρ=43.64% **π=57.59%** **ρ=37.26%** 2744	π=55.42% ρ=44.09% **π=59.11%** **ρ=37.65%** 2672	π=58.84% ρ=47.8% **π=61.86%** **ρ=38.15%** 3103	**π=62.23%** **ρ=56.04%** π=60.5% ρ=35.27% 3879
PredLink Paragraph	**π=79.25%** **ρ=76.33%** π=78.15% ρ=72.95% 3667	**π=64.68%** **ρ=56.27%** π=61.89% ρ=52.07% 3655	**π=60.3%** **ρ=44.85%** π=56.74% ρ=38.36% 3013	π=58.84% ρ=47.8% **π=61.86%** **ρ=38.15%** 3103	**π=64.88%** **ρ=51.51%** π=63.3% ρ=41.72% 2715	π=63.93% ρ=59.08% **π=65.51%** **ρ=45.06%** 3882
Own Text	**π=79.58%** **ρ=77.93%** π=78.69% ρ=73.23% 3898	**π=67.81%** **ρ=60.71%** π=59.24% ρ=48.26% 3898	**π=60.5%** **ρ=54.57%** π=59.29% ρ=34.18% 3864	**π=62.23%** **ρ=56.04%** π=60.5% ρ=35.27% 3879	π=63.93% ρ=59.08% **π=65.51%** **ρ=45.06%** 3882	**π=56.47%** **ρ=49.71%** π=- ρ=- 3831

Table B.4: Alleskar Merging Round Robin Meta Predecessor -Alleskar Merging Round Robin Hyperlink Ensembles
Meta Predecessor outperforms Hyperlink Ensembles in almost all the cases. Only 4 combinations see the Hyperlink Ensembles have a better precision, mostly with *PredHeadings*.

B.1.2 WebKB

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=41.07%** **ρ=17.94%** π=36.85% ρ=18.48% 3006	**π=56.66%** **ρ=20.35%** π=52.46% ρ=20.98% 3016	π=30.13% ρ=16.91% **π=33.14%** **ρ=18.3%** 3007	**π=36.49%** **ρ=17.36%** π=26.85% ρ=16.48% 3016	π=35.51% ρ=19.08% **π=39.79%** **ρ=19.01%** 3011	**π=44.27%** **ρ=24.31%** π=40.76% ρ=22.45% 8276
Pred LinkTags	**π=56.66%** **ρ=20.35%** π=52.46% ρ=20.98% 3016	π=35.54% ρ=21.35% **π=41.99%** **ρ=27.35%** 2940	π=34.02% ρ=19% **π=47.3%** **ρ=24.95%** 2941	π=29.23% ρ=17.36% **π=33.09%** **ρ=18.49%** 3001	π=30.53% ρ=19.87% **π=34.51%** **ρ=19.58%** 2954	π=43.23% ρ=24.44% **π=44.31%** **ρ=22.63%** 8276
PredList Headings	π=30.13% ρ=16.91% **π=33.14%** **ρ=18.3%** 3007	π=34.02% ρ=19% **π=47.3%** **ρ=24.95%** 2941	π=17.38% ρ=14.89% **π=24.39%** **ρ=15.9%** 1644	π=27.86% ρ=17.3% **π=30.09%** **ρ=18.91%** 2832	π=26.14% ρ=16.65% **π=27.82%** **ρ=16.2%** 2402	**π=43.71%** **ρ=24.02%** π=40.46% ρ=23.28% 8276
Pred Headings	**π=36.49%** **ρ=17.36%** π=26.85% ρ=16.48% 3016	π=29.23% ρ=17.36% **π=33.09%** **ρ=18.49%** 3001	π=27.86% ρ=17.3% **π=30.09%** **ρ=18.91%** 2832	**π=28.35%** **ρ=17.37%** π=20.32% ρ=15.7% 2828	π=26.13% ρ=16.84% **π=26.82%** **ρ=16.89%** 2911	**π=43.96%** **ρ=23.65%** π=36.67% ρ=19.26% 8276
PredLink Paragraph	π=35.51% ρ=19.08% **π=39.79%** **ρ=19.01%** 3011	π=30.53% ρ=19.87% **π=34.51%** **ρ=19.58%** 2954	π=26.14% ρ=16.65% **π=27.82%** **ρ=16.2%** 2402	π=26.13% ρ=16.84% **π=26.82%** **ρ=16.89%** 2911	π=29.17% ρ=16.71% **π=29.23%** **ρ=18.03%** 1143	**π=43.5%** **ρ=24.69%** π=41.65% ρ=24.02% 8276
Own Text	**π=44.27%** **ρ=24.31%** π=40.76% ρ=22.45% 8276	π=43.23% ρ=24.44% **π=44.31%** **ρ=22.63%** 8276	**π=43.71%** **ρ=24.02%** π=40.46% ρ=23.28% 8276	**π=43.96%** **ρ=23.65%** π=36.67% ρ=19.26% 8276	**π=43.5%** **ρ=24.69%** π=41.65% ρ=24.02% 8276	**π=45.37%** **ρ=24.71%** π=- ρ=- 8276

Table B.5: WebKB Tagging One Against All Meta Predecessor -WebKB Tagging One Against All Hyperlink Ensembles
Hyperlink Ensembles outperforms Meta Predecessor with PredLinkTags, PredListHeadings and PredLinkParagraph.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=41.07%** **ρ=17.94%** π=36.85% ρ=18.48% 3006	**π=44.4%** **ρ=21.05%** π=39.09% ρ=19.59% 3016	π=28.08% ρ=15.3% **π=29.91%** **ρ=15.19%** 3007	**π=37.51%** **ρ=16.95%** π=16.02% ρ=15.45% 3016	**π=42.74%** **ρ=19.43%** π=40.71% ρ=19.05% 3011	**π=40.45%** **ρ=21.79%** π=17.88% ρ=14.77% 8276
Pred LinkTags	**π=44.4%** **ρ=21.05%** π=39.09% ρ=19.59% 3016	π=35.54% ρ=21.35% **π=41.99%** **ρ=27.35%** 2940	π=23.48% ρ=16.11% **π=36.74%** **ρ=23.47%** 2941	π=32.96% ρ=16.34% **π=36.96%** **ρ=17.8%** 3001	π=30.5% ρ=20.24% **π=34.4%** **ρ=20.16%** 2954	**π=43.01%** **ρ=23.82%** π=39.31% ρ=21.09% 8276
PredList Headings	π=28.08% ρ=15.3% **π=29.91%** **ρ=15.19%** 3007	π=23.48% ρ=16.11% **π=36.74%** **ρ=23.47%** 2941	π=17.38% ρ=14.89% **π=24.39%** **ρ=15.9%** 1644	**π=30.41%** **ρ=17.71%** π=25.56% ρ=16.95% 2832	π=19.99% ρ=14.9% **π=23.98%** **ρ=15.39%** 2402	**π=42.29%** **ρ=23.29%** π=14.66% ρ=15% 8276
Pred Headings	**π=37.51%** **ρ=16.95%** π=16.02% ρ=15.45% 3016	π=32.96% ρ=16.34% **π=36.96%** **ρ=17.8%** 3001	**π=30.41%** **ρ=17.71%** π=25.56% ρ=16.95% 2832	**π=28.35%** **ρ=17.37%** π=20.32% ρ=15.7% 2828	**π=26.55%** **ρ=16.73%** π=20.93% ρ=15.84% 2911	**π=42.83%** **ρ=23.12%** π=17.16% ρ=14.56% 8276
PredLink Paragraph	**π=42.74%** **ρ=19.43%** π=40.71% ρ=19.05% 3011	π=30.5% ρ=20.24% **π=34.4%** **ρ=20.16%** 2954	π=19.99% ρ=14.9% **π=23.98%** **ρ=15.39%** 2402	**π=26.55%** **ρ=16.73%** π=20.93% ρ=15.84% 2911	π=29.17% ρ=16.71% **π=29.23%** **ρ=18.03%** 1143	**π=42.45%** **ρ=23.76%** π=25.35% ρ=16.53% 8276
Own Text	**π=40.45%** **ρ=21.79%** π=17.88% ρ=14.77% 8276	**π=43.01%** **ρ=23.82%** π=39.31% ρ=21.09% 8276	**π=42.29%** **ρ=23.29%** π=14.66% ρ=15% 8276	**π=42.83%** **ρ=23.12%** π=17.16% ρ=14.56% 8276	**π=42.45%** **ρ=23.76%** π=25.35% ρ=16.53% 8276	**π=45.37%** **ρ=24.71%** π=- ρ=- 8276

Table B.6: WebKB Merging One Against All Meta Predecessor -WebKB Merging One Against All Hyperlink Ensembles
Hyperlink Ensembles outperforms Meta Predecessor with PredLinkTags and PredList-Headings.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=40.08%** **ρ=19.46%** π=39.56% ρ=20.52% 3006	π=49.43% ρ=21.95% **π=51.6%** **ρ=24.26%** 3016	**π=34.29%** **ρ=17.96%** π=32.95% ρ=19.99% 3007	**π=36.96%** **ρ=18.13%** π=27.39% ρ=17.03% 3016	π=39.29% ρ=19.89% **π=40.38%** **ρ=22.64%** 3011	**π=42.27%** **ρ=28.96%** π=37.23% ρ=25.37% 8276
Pred LinkTags	π=49.43% ρ=21.95% **π=51.6%** **ρ=24.26%** 3016	π=34.16% ρ=21.86% **π=41.23%** **ρ=29.74%** 2940	π=35.53% ρ=19.86% **π=39.01%** **ρ=26.41%** 2941	π=27.21% ρ=17.85% **π=30.63%** **ρ=19.05%** 3001	π=29.66% ρ=19.89% **π=33.85%** **ρ=22.74%** 2954	**π=42.76%** **ρ=29.18%** π=39.78% ρ=24.96% 8276
PredList Headings	**π=34.29%** **ρ=17.96%** π=32.95% ρ=19.99% 3007	π=35.53% ρ=19.86% **π=39.01%** **ρ=26.41%** 2941	**π=30.7%** **ρ=19.42%** π=24.44% ρ=17.22% 1644	π=25.11% ρ=17.58% **π=26.04%** **ρ=17.45%** 2832	π=25.7% ρ=17.18% **π=26.04%** **ρ=16.54%** 2402	**π=41.85%** **ρ=28.53%** π=38.55% ρ=26.67% 8276
Pred Headings	**π=36.96%** **ρ=18.13%** π=27.39% ρ=17.03% 3016	π=27.21% ρ=17.85% **π=30.63%** **ρ=19.05%** 3001	π=25.11% ρ=17.58% **π=26.04%** **ρ=17.45%** 2832	**π=25.62%** **ρ=16.64%** π=22.8% ρ=16.05% 2828	**π=24.5%** **ρ=17.1%** π=24.46% ρ=16.9% 2911	**π=41.72%** **ρ=28.36%** π=35.23% ρ=21.44% 8276
PredLink Paragraph	π=39.29% ρ=19.89% **π=40.38%** **ρ=22.64%** 3011	π=29.66% ρ=19.89% **π=33.85%** **ρ=22.74%** 2954	π=25.7% ρ=17.18% **π=26.04%** **ρ=16.54%** 2402	**π=24.5%** **ρ=17.1%** π=24.46% ρ=16.9% 2911	π=27.79% ρ=16.86% **π=28.86%** **ρ=19.2%** 1143	π=41.51% ρ=28.86% **π=42.39%** **ρ=28.29%** 8276
Own Text	**π=42.27%** **ρ=28.96%** π=37.23% ρ=25.37% 8276	**π=42.76%** **ρ=29.18%** π=39.78% ρ=24.96% 8276	**π=41.85%** **ρ=28.53%** π=38.55% ρ=26.67% 8276	**π=41.72%** **ρ=28.36%** π=35.23% ρ=21.44% 8276	π=41.51% ρ=28.86% **π=42.39%** **ρ=28.29%** 8276	**π=42%** **ρ=29.13%** π=- ρ=- 8276

Table B.7: WebKB Tagging Round Robin Meta Predecessor -WebKB Tagging Round Robin Hyperlink Ensembles
Hyperlink Ensembles outperforms Meta Predecessor with PredLinkTags, PredListHeadings and PredLinkParagraph.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=40.08%** **ρ=19.46%** π=39.56% ρ=20.52% 3006	**π=41.85%** **ρ=22.57%** π=39.63% ρ=22.72% 3016	**π=30.79%** **ρ=16.08%** π=28.91% ρ=18.2% 3007	**π=38.14%** **ρ=17.37%** π=23.98% ρ=16.18% 3016	**π=41.58%** **ρ=20.81%** π=39.82% ρ=22.76% 3011	**π=40.1%** **ρ=26.22%** π=18.88% ρ=15.4% 8276
Pred LinkTags	**π=41.85%** **ρ=22.57%** π=39.63% ρ=22.72% 3016	π=34.16% ρ=21.86% **π=41.23%** **ρ=29.74%** 2940	π=24.26% ρ=16.6% **π=34.83%** **ρ=25.3%** 2941	π=30.5% ρ=17.6% **π=32.75%** **ρ=18.81%** 3001	π=31.18% ρ=20.56% **π=36.25%** **ρ=22.26%** 2954	**π=41.63%** **ρ=27.88%** π=39.87% ρ=21.47% 8276
PredList Headings	**π=30.79%** **ρ=16.08%** π=28.91% ρ=18.2% 3007	π=24.26% ρ=16.6% **π=34.83%** **ρ=25.3%** 2941	**π=30.7%** **ρ=19.42%** π=24.44% ρ=17.22% 1644	π=27.63% ρ=17.29% **π=30.31%** **ρ=18.64%** 2832	**π=28.76%** **ρ=16.03%** π=24.18% ρ=15.73% 2402	**π=39.42%** **ρ=26.96%** π=13.71% ρ=14.99% 8276
Pred Headings	**π=38.14%** **ρ=17.37%** π=23.98% ρ=16.18% 3016	π=30.5% ρ=17.6% **π=32.75%** **ρ=18.81%** 3001	π=27.63% ρ=17.29% **π=30.31%** **ρ=18.64%** 2832	**π=25.62%** **ρ=16.64%** π=22.8% ρ=16.05% 2828	π=26.62% ρ=17.19% **π=29%** **ρ=17.17%** 2911	**π=40.76%** **ρ=27.28%** π=14.97% ρ=14.5% 8276
PredLink Paragraph	**π=41.58%** **ρ=20.81%** π=39.82% ρ=22.76% 3011	π=31.18% ρ=20.56% **π=36.25%** **ρ=22.26%** 2954	**π=28.76%** **ρ=16.03%** π=24.18% ρ=15.73% 2402	π=26.62% ρ=17.19% **π=29%** **ρ=17.17%** 2911	π=27.79% ρ=16.86% **π=28.86%** **ρ=19.2%** 1143	**π=41.02%** **ρ=28.37%** π=26.25% ρ=17.52% 8276
Own Text	**π=40.1%** **ρ=26.22%** π=18.88% ρ=15.4% 8276	**π=41.63%** **ρ=27.88%** π=39.87% ρ=21.47% 8276	**π=39.42%** **ρ=26.96%** π=13.71% ρ=14.99% 8276	**π=40.76%** **ρ=27.28%** π=14.97% ρ=14.5% 8276	**π=41.02%** **ρ=28.37%** π=26.25% ρ=17.52% 8276	**π=42%** **ρ=29.13%** π=- ρ=- 8276

Table B.8: WebKB Merging Round Robin Meta Predecessor -WebKB Merging Round Robin Hyperlink Ensembles
Hyperlink Ensembles outperforms Meta Predecessor with PredLinkTags and PredList-Headings.

B.2 Meta learned Hyperlink Ensembles and Hyperlink Ensembles

Green if for Meta Learned Hyperlink Ensembles and blue for Hyperlink Ensembles

B.2.1 Alleskalr

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=79.61% ρ=56.5% **π=82.97%** **ρ=46.92%** 3664	π=80.26% ρ=55.67% **π=83.41%** **ρ=47.31%** 3678	π=78.87% ρ=55.71% **π=83.06%** **ρ=47.07%** 3665	π=77.29% ρ=52.49% **π=82.89%** **ρ=38.34%** 3665	π=77.66% ρ=53.38% **π=83.25%** **ρ=45.75%** 3667	π=76.24% ρ=50.33% **π=81.88%** **ρ=39.39%** 3898
Pred LinkTags	π=80.26% ρ=55.67% **π=83.41%** **ρ=47.31%** 3678	**π=77.47%** **ρ=37.71%** π=75.82% ρ=37.58% 3653	π=76.67% ρ=36.16% **π=76.67%** **ρ=35.77%** 3653	**π=73.06%** **ρ=31.9%** π=72.03% ρ=29.86% 3653	π=71.31% ρ=32.75% **π=75.75%** **ρ=31.4%** 3655	π=67.63% ρ=33.35% **π=73.92%** **ρ=30.45%** 3898
PredList Headings	π=78.87% ρ=55.71% **π=83.06%** **ρ=47.07%** 3665	π=76.67% ρ=36.16% **π=76.67%** **ρ=35.77%** 3653	π=64.67% ρ=27.62% **π=68.19%** **ρ=28.54%** 1870	π=48.99% ρ=29.51% **π=67.81%** **ρ=28.37%** 2744	π=75.82% ρ=26.26% **π=77.62%** **ρ=27.61%** 3013	π=69.92% ρ=32.83% **π=71.62%** **ρ=27.81%** 3864
Pred Headings	π=77.29% ρ=52.49% **π=82.89%** **ρ=38.34%** 3665	**π=73.06%** **ρ=31.9%** π=72.03% ρ=29.86% 3653	π=48.99% ρ=29.51% **π=67.81%** **ρ=28.37%** 2744	π=56.26% ρ=31.59% **π=66.41%** **ρ=29.12%** 2672	π=62.95% ρ=26.17% **π=70.52%** **ρ=26.91%** 3103	π=66.53% ρ=31.9% **π=77.91%** **ρ=25.34%** 3879
PredLink Paragraph	π=77.66% ρ=53.38% **π=83.25%** **ρ=45.75%** 3667	π=71.31% ρ=32.75% **π=75.75%** **ρ=31.4%** 3655	π=75.82% ρ=26.26% **π=77.62%** **ρ=27.61%** 3013	π=62.95% ρ=26.17% **π=70.52%** **ρ=26.91%** 3103	π=74.32% ρ=32.89% **π=74.94%** **ρ=30.62%** 2715	π=69.42% ρ=34.14% **π=74.32%** **ρ=28.61%** 3882
Own Text	π=76.24% ρ=50.33% **π=81.88%** **ρ=39.39%** 3898	π=67.63% ρ=33.35% **π=73.92%** **ρ=30.45%** 3898	π=69.92% ρ=32.83% **π=71.62%** **ρ=27.81%** 3864	π=66.53% ρ=31.9% **π=77.91%** **ρ=25.34%** 3879	π=69.42% ρ=34.14% **π=74.32%** **ρ=28.61%** 3882	π=- ρ=- π=- ρ=- 3831

Table B.9: Alleskalr Tagging One Against All Meta learned Hyperlink Ensembles -Alleskalr Tagging One Against All Hyperlink Ensembles
The two only cases where Meta learned Hyperlink Ensembles outperforms Hyperlink ensembles (PredLinkTags and PredLinkTags&PredHeadings) are combinations with features gathering few words. In this case, merging those few features favorises the learning phase.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=79.61% ρ=56.5% **π=82.97%** **ρ=46.92%** 3664	π=80.68% ρ=55.84% **π=84.23%** **ρ=51.09%** 3678	π=79.03% ρ=56.44% **π=83.08%** **ρ=46.96%** 3665	π=78.04% ρ=52.14% **π=83.09%** **ρ=38.21%** 3665	π=79.86% ρ=55.36% **π=83.43%** **ρ=48.43%** 3667	π=80.63% ρ=57.22% **π=83.24%** **ρ=43.5%** 3898
Pred LinkTags	π=80.68% ρ=55.84% **π=84.23%** **ρ=51.09%** 3678	**π=77.47%** **ρ=37.71%** π=75.82% ρ=37.58% 3653	π=75.28% ρ=36.28% **π=77.05%** **ρ=35.79%** 3653	π=73.7% ρ=33.75% **π=76.87%** **ρ=31.86%** 3653	π=76.3% ρ=37.75% **π=77.71%** **ρ=35.36%** 3655	π=73.83% ρ=36.93% **π=76.46%** **ρ=33.01%** 3898
PredList Headings	π=79.03% ρ=56.44% **π=83.08%** **ρ=46.96%** 3665	π=75.28% ρ=36.28% **π=77.05%** **ρ=35.79%** 3653	π=64.67% ρ=27.62% **π=68.19%** **ρ=28.54%** 1870	π=52.16% ρ=29.59% **π=66.75%** **ρ=27.95%** 2744	π=76.44% ρ=27.53% **π=78.48%** **ρ=28.62%** 3013	π=69.4% ρ=32.12% **π=75.07%** **ρ=26.79%** 3864
Pred Headings	π=78.04% ρ=52.14% **π=83.09%** **ρ=38.21%** 3665	π=73.7% ρ=33.75% **π=76.87%** **ρ=31.86%** 3653	π=52.16% ρ=29.59% **π=66.75%** **ρ=27.95%** 2744	π=56.26% ρ=31.59% **π=66.41%** **ρ=29.12%** 2672	π=57.94% ρ=28.39% **π=71.88%** **ρ=25.87%** 3103	π=67.25% ρ=31.5% **π=72.64%** **ρ=25.14%** 3879
PredLink Paragraph	π=79.86% ρ=55.36% **π=83.43%** **ρ=48.43%** 3667	π=76.3% ρ=37.75% **π=77.71%** **ρ=35.36%** 3655	π=76.44% ρ=27.53% **π=78.48%** **ρ=28.62%** 3013	π=57.94% ρ=28.39% **π=71.88%** **ρ=25.87%** 3103	π=74.32% ρ=32.89% **π=74.94%** **ρ=30.62%** 2715	π=75.72% ρ=36.04% **π=78.35%** **ρ=30.39%** 3882
Own Text	π=80.63% ρ=57.22% **π=83.24%** **ρ=43.5%** 3898	π=73.83% ρ=36.93% **π=76.46%** **ρ=33.01%** 3898	π=69.4% ρ=32.12% **π=75.07%** **ρ=26.79%** 3864	π=67.25% ρ=31.5% **π=72.64%** **ρ=25.14%** 3879	π=75.72% ρ=36.04% **π=78.35%** **ρ=30.39%** 3882	π=- ρ=- π=- ρ=- 3831

Table B.10: Alleskiar Merging One Against All Meta learned Hyperlink Ensembles - Alleskiar Merging One Against All Hyperlink Ensembles
The only case where Meta learned Hyperlink Ensembles outperform Hyperlink ensembles (PredLinkTags alone) is with a feature location which gathers few words. In this case, merging those few features favorises the learning phase.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=74.2% ρ=71.23% **π=77.83%** **ρ=72.85%** 3664	π=74.91% ρ=71.56% **π=77.59%** **ρ=73.19%** 3678	π=74.08% ρ=71.48% **π=77.32%** **ρ=72.06%** 3665	π=72.18% ρ=68.97% **π=75.58%** **ρ=67.67%** 3665	π=71.57% ρ=67.96% **π=76.66%** **ρ=70.12%** 3667	π=69.74% ρ=66.7% **π=73.67%** **ρ=65.76%** 3898
Pred LinkTags	π=74.91% ρ=71.56% **π=77.59%** **ρ=73.19%** 3678	π=56.81% ρ=49.16% **π=59.15%** **ρ=48.9%** 3653	π=56.51% ρ=44.53% **π=56.62%** **ρ=47.74%** 3653	π=48.95% ρ=41.03% **π=56.91%** **ρ=40.13%** 3653	π=49.36% ρ=42.24% **π=55.31%** **ρ=44.83%** 3655	**π=56.33%** **ρ=46.86%** π=55% ρ=44.37% 3898
PredList Headings	π=74.08% ρ=71.48% **π=77.32%** **ρ=72.06%** 3665	π=56.51% ρ=44.53% **π=56.62%** **ρ=47.74%** 3653	π=45.49% ρ=36.53% **π=47.69%** **ρ=33.97%** 1870	π=47.29% ρ=35.45% **π=56.71%** **ρ=37%** 2744	π=53.83% ρ=39.68% **π=56.28%** **ρ=37.08%** 3013	**π=55.66%** **ρ=48.7%** π=54.29% ρ=42.65% 3864
Pred Headings	π=72.18% ρ=68.97% **π=75.58%** **ρ=67.67%** 3665	π=48.95% ρ=41.03% **π=56.91%** **ρ=40.13%** 3653	π=47.29% ρ=35.45% **π=56.71%** **ρ=37%** 2744	π=48.03% ρ=37.94% **π=59.11%** **ρ=37.65%** 2672	π=51.27% ρ=38.06% **π=61.39%** **ρ=38.76%** 3103	**π=56.91%** **ρ=46.94%** π=56.34% ρ=38.08% 3879
PredLink Paragraph	π=71.57% ρ=67.96% **π=76.66%** **ρ=70.12%** 3667	π=49.36% ρ=42.24% **π=55.31%** **ρ=44.83%** 3655	π=53.83% ρ=39.68% **π=56.28%** **ρ=37.08%** 3013	π=51.27% ρ=38.06% **π=61.39%** **ρ=38.76%** 3103	π=57.42% ρ=44.6% **π=63.3%** **ρ=41.72%** 2715	π=55.56% ρ=51.3% **π=59.7%** **ρ=42.89%** 3882
Own Text	π=69.74% ρ=66.7% **π=73.67%** **ρ=65.76%** 3898	**π=56.33%** **ρ=46.86%** π=55% ρ=44.37% 3898	**π=55.66%** **ρ=48.7%** π=54.29% ρ=42.65% 3864	**π=56.91%** **ρ=46.94%** π=56.34% ρ=38.08% 3879	π=55.56% ρ=51.3% **π=59.7%** **ρ=42.89%** 3882	π=- ρ=- π=- ρ=- 3831

Table B.11: Alleskla Tagging Round Robin Meta learned Hyperlink Ensembles -Alleskla Tagging Round Robin Hyperlink Ensembles
In all the purely non-local features combinations, Hyperlink Ensembles outperforms Meta Learned Hyperlink Ensembles.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=74.2% ρ=71.23% **π=77.83%** **ρ=72.85%** 3664	π=75.63% ρ=71.75% **π=78.87%** **ρ=74.84%** 3678	π=73.79% ρ=71.08% **π=77.76%** **ρ=72.58%** 3665	π=72.56% ρ=69.22% **π=76.14%** **ρ=67.9%** 3665	π=73.61% ρ=70.97% **π=78.15%** **ρ=72.95%** 3667	π=75.58% ρ=72.99% **π=78.69%** **ρ=73.23%** 3898
Pred LinkTags	π=75.63% ρ=71.75% **π=78.87%** **ρ=74.84%** 3678	π=56.81% ρ=49.16% **π=59.15%** **ρ=48.9%** 3653	π=55.81% ρ=46% **π=57.45%** **ρ=48.66%** 3653	π=52.69% ρ=43.49% **π=57.72%** **ρ=41.71%** 3653	π=58.72% ρ=47.83% **π=61.89%** **ρ=52.07%** 3655	**π=61.51%** **ρ=53.14%** π=59.24% ρ=48.26% 3898
PredList Headings	π=73.79% ρ=71.08% **π=77.76%** **ρ=72.58%** 3665	π=55.81% ρ=46% **π=57.45%** **ρ=48.66%** 3653	π=45.49% ρ=36.53% **π=47.69%** **ρ=33.97%** 1870	π=46.96% ρ=38.72% **π=57.59%** **ρ=37.26%** 2744	π=55.02% ρ=40.82% **π=56.74%** **ρ=38.36%** 3013	π=56.64% ρ=51.12% **π=59.29%** **ρ=34.18%** 3864
Pred Headings	π=72.56% ρ=69.22% **π=76.14%** **ρ=67.9%** 3665	π=52.69% ρ=43.49% **π=57.72%** **ρ=41.71%** 3653	π=46.96% ρ=38.72% **π=57.59%** **ρ=37.26%** 2744	π=48.03% ρ=37.94% **π=59.11%** **ρ=37.65%** 2672	π=53.17% ρ=41.05% **π=61.86%** **ρ=38.15%** 3103	π=56.59% ρ=46.54% **π=60.5%** **ρ=35.27%** 3879
PredLink Paragraph	π=73.61% ρ=70.97% **π=78.15%** **ρ=72.95%** 3667	π=58.72% ρ=47.83% **π=61.89%** **ρ=52.07%** 3655	π=55.02% ρ=40.82% **π=56.74%** **ρ=38.36%** 3013	π=53.17% ρ=41.05% **π=61.86%** **ρ=38.15%** 3103	π=57.42% ρ=44.6% **π=63.3%** **ρ=41.72%** 2715	π=61.63% ρ=55.12% **π=65.51%** **ρ=45.06%** 3882
Own Text	π=75.58% ρ=72.99% **π=78.69%** **ρ=73.23%** 3898	**π=61.51%** **ρ=53.14%** π=59.24% ρ=48.26% 3898	π=56.64% ρ=51.12% **π=59.29%** **ρ=34.18%** 3864	π=56.59% ρ=46.54% **π=60.5%** **ρ=35.27%** 3879	π=61.63% ρ=55.12% **π=65.51%** **ρ=45.06%** 3882	π=- ρ=- π=- ρ=- 3831

Table B.12: -Allesklar Merging Round Robin Hyperlink Ensembles Hyperlink Ensembles outperforms Meta Learned Hyperlink Ensembles

B.2.2 WebKB

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=41.51%** **ρ=19.78%** π=36.85% ρ=18.48% 3006	π=51.68% ρ=21.36% **π=52.46%** **ρ=20.98%** 3016	π=29.33% ρ=17.61% **π=33.14%** **ρ=18.3%** 3007	**π=34.39%** **ρ=19.25%** π=26.85% ρ=16.48% 3016	π=35.79% ρ=22.49% **π=39.79%** **ρ=19.01%** 3011	**π=41.78%** **ρ=23.5%** π=40.76% ρ=22.45% 8276
Pred LinkTags	π=51.68% ρ=21.36% **π=52.46%** **ρ=20.98%** 3016	π=32.51% ρ=20.5% **π=41.99%** **ρ=27.35%** 2940	π=31.33% ρ=18.44% **π=47.3%** **ρ=24.95%** 2941	**π=33.91%** **ρ=19.7%** π=33.09% ρ=18.49% 3001	π=30.16% ρ=21.02% **π=34.51%** **ρ=19.58%** 2954	π=41.96% ρ=23.64% **π=44.31%** **ρ=22.63%** 8276
PredList Headings	π=29.33% ρ=17.61% **π=33.14%** **ρ=18.3%** 3007	π=31.33% ρ=18.44% **π=47.3%** **ρ=24.95%** 2941	π=16.45% ρ=14.94% **π=24.39%** **ρ=15.9%** 1644	π=26.96% ρ=18.67% **π=30.09%** **ρ=18.91%** 2832	π=27.25% ρ=17.97% **π=27.82%** **ρ=16.2%** 2402	**π=42.33%** **ρ=23.88%** π=40.46% ρ=23.28% 8276
Pred Headings	**π=34.39%** **ρ=19.25%** π=26.85% ρ=16.48% 3016	**π=33.91%** **ρ=19.7%** π=33.09% ρ=18.49% 3001	π=26.96% ρ=18.67% **π=30.09%** **ρ=18.91%** 2832	**π=24.87%** **ρ=18.28%** π=20.32% ρ=15.7% 2828	**π=27.28%** **ρ=18.8%** π=26.82% ρ=16.89% 2911	**π=41.1%** **ρ=22.86%** π=36.67% ρ=19.26% 8276
PredLink Paragraph	π=35.79% ρ=22.49% **π=39.79%** **ρ=19.01%** 3011	π=30.16% ρ=21.02% **π=34.51%** **ρ=19.58%** 2954	π=27.25% ρ=17.97% **π=27.82%** **ρ=16.2%** 2402	**π=27.28%** **ρ=18.8%** π=26.82% ρ=16.89% 2911	**π=29.74%** **ρ=17.02%** π=29.23% ρ=18.03% 1143	**π=42.31%** **ρ=24.53%** π=41.65% ρ=24.02% 8276
Own Text	**π=41.78%** **ρ=23.5%** π=40.76% ρ=22.45% 8276	π=41.96% ρ=23.64% **π=44.31%** **ρ=22.63%** 8276	**π=42.33%** **ρ=23.88%** π=40.46% ρ=23.28% 8276	**π=41.1%** **ρ=22.86%** π=36.67% ρ=19.26% 8276	**π=42.31%** **ρ=24.53%** π=41.65% ρ=24.02% 8276	π=- ρ=- π=- ρ=- 8276

Table B.13: WebKB Tagging One Against All Meta learned Hyperlink Ensembles -WebKB Tagging One Against All Hyperlink Ensembles
Meta learned Hyperlink Ensembles outperforms Hyperlinks Ensembles with PredHeadings and with the combinations which include OwnText.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=41.51%** **ρ=19.78%** π=36.85% ρ=18.48% 3006	**π=44.54%** **ρ=22.3%** π=39.09% ρ=19.59% 3016	π=24.85% ρ=15.35% **π=29.91%** **ρ=15.19%** 3007	**π=33.12%** **ρ=17.71%** π=16.02% ρ=15.45% 3016	**π=42.41%** **ρ=22.83%** π=40.71% ρ=19.05% 3011	**π=28.21%** **ρ=25.54%** π=17.88% ρ=14.77% 8276
Pred LinkTags	**π=44.54%** **ρ=22.3%** π=39.09% ρ=19.59% 3016	π=32.51% ρ=20.5% **π=41.99%** **ρ=27.35%** 2940	π=24.41% ρ=16.48% **π=36.74%** **ρ=23.47%** 2941	π=30.52% ρ=20.71% **π=36.96%** **ρ=17.8%** 3001	π=30.82% ρ=21.37% **π=34.4%** **ρ=20.16%** 2954	π=31.13% ρ=26.66% **π=39.31%** **ρ=21.09%** 8276
PredList Headings	π=24.85% ρ=15.35% **π=29.91%** **ρ=15.19%** 3007	π=24.41% ρ=16.48% **π=36.74%** **ρ=23.47%** 2941	π=16.45% ρ=14.94% **π=24.39%** **ρ=15.9%** 1644	π=20.89% ρ=16.96% **π=25.56%** **ρ=16.95%** 2832	π=18.8% ρ=15.01% **π=23.98%** **ρ=15.39%** 2402	**π=33.51%** **ρ=31.88%** π=14.66% ρ=15% 8276
Pred Headings	**π=33.12%** **ρ=17.71%** π=16.02% ρ=15.45% 3016	π=30.52% ρ=20.71% **π=36.96%** **ρ=17.8%** 3001	π=20.89% ρ=16.96% **π=25.56%** **ρ=16.95%** 2832	**π=24.87%** **ρ=18.28%** π=20.32% ρ=15.7% 2828	**π=28.89%** **ρ=18.65%** π=20.93% ρ=15.84% 2911	**π=29.21%** **ρ=22.02%** π=17.16% ρ=14.56% 8276
PredLink Paragraph	**π=42.41%** **ρ=22.83%** π=40.71% ρ=19.05% 3011	π=30.82% ρ=21.37% **π=34.4%** **ρ=20.16%** 2954	π=18.8% ρ=15.01% **π=23.98%** **ρ=15.39%** 2402	**π=28.89%** **ρ=18.65%** π=20.93% ρ=15.84% 2911	**π=29.74%** **ρ=17.02%** π=29.23% ρ=18.03% 1143	**π=30.21%** **ρ=26%** π=25.35% ρ=16.53% 8276
Own Text	**π=28.21%** **ρ=25.54%** π=17.88% ρ=14.77% 8276	π=31.13% ρ=26.66% **π=39.31%** **ρ=21.09%** 8276	**π=33.51%** **ρ=31.88%** π=14.66% ρ=15% 8276	**π=29.21%** **ρ=22.02%** π=17.16% ρ=14.56% 8276	**π=30.21%** **ρ=26%** π=25.35% ρ=16.53% 8276	π=- ρ=- π=- ρ=- 8276

Table B.14: WebKB Merging One Against All Meta learned Hyperlink Ensembles -WebKB Merging One Against All Hyperlink Ensembles
With merging, Meta learned Hyperlink Ensembles outperforms Hyperlink Ensembles with WordsAround, PredHeadings and OwnText.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=39.95%** **ρ=22.75%** π=39.56% ρ=20.52% 3006	π=44.07% ρ=24.52% **π=51.6%** **ρ=24.26%** 3016	π=32.25% ρ=18.79% **π=32.95%** **ρ=19.99%** 3007	**π=33.18%** **ρ=20.62%** π=27.39% ρ=17.03% 3016	π=39.72% ρ=23.52% **π=40.38%** **ρ=22.64%** 3011	**π=39.84%** **ρ=27.54%** π=37.23% ρ=25.37% 8276
Pred LinkTags	π=44.07% ρ=24.52% **π=51.6%** **ρ=24.26%** 3016	π=32.63% ρ=21.17% **π=41.23%** **ρ=29.74%** 2940	π=36.03% ρ=19.62% **π=39.01%** **ρ=26.41%** 2941	**π=32.68%** **ρ=21.57%** π=30.63% ρ=19.05% 3001	π=29.64% ρ=21.24% **π=33.85%** **ρ=22.74%** 2954	**π=45.06%** **ρ=28.97%** π=39.78% ρ=24.96% 8276
PredList Headings	π=32.25% ρ=18.79% **π=32.95%** **ρ=19.99%** 3007	π=36.03% ρ=19.62% **π=39.01%** **ρ=26.41%** 2941	**π=32.32%** **ρ=21.21%** π=24.44% ρ=17.22% 1644	π=25.11% ρ=18.65% **π=26.04%** **ρ=17.45%** 2832	**π=31.84%** **ρ=19.44%** π=26.04% ρ=16.54% 2402	**π=41.01%** **ρ=27.94%** π=38.55% ρ=26.67% 8276
Pred Headings	**π=33.18%** **ρ=20.62%** π=27.39% ρ=17.03% 3016	**π=32.68%** **ρ=21.57%** π=30.63% ρ=19.05% 3001	π=25.11% ρ=18.65% **π=26.04%** **ρ=17.45%** 2832	**π=24.67%** **ρ=18.53%** π=22.8% ρ=16.05% 2828	**π=27.23%** **ρ=19.06%** π=24.46% ρ=16.9% 2911	**π=39.48%** **ρ=27.03%** π=35.23% ρ=21.44% 8276
PredLink Paragraph	π=39.72% ρ=23.52% **π=40.38%** **ρ=22.64%** 3011	π=29.64% ρ=21.24% **π=33.85%** **ρ=22.74%** 2954	**π=31.84%** **ρ=19.44%** π=26.04% ρ=16.54% 2402	**π=27.23%** **ρ=19.06%** π=24.46% ρ=16.9% 2911	π=28.25% ρ=17.15% **π=28.86%** **ρ=19.2%** 1143	π=41.15% ρ=28.63% **π=42.39%** **ρ=28.29%** 8276
Own Text	**π=39.84%** **ρ=27.54%** π=37.23% ρ=25.37% 8276	**π=45.06%** **ρ=28.97%** π=39.78% ρ=24.96% 8276	**π=41.01%** **ρ=27.94%** π=38.55% ρ=26.67% 8276	**π=39.48%** **ρ=27.03%** π=35.23% ρ=21.44% 8276	π=41.15% ρ=28.63% **π=42.39%** **ρ=28.29%** 8276	π=- ρ=- π=- ρ=- 8276

Table B.15: WebKB Tagging Round Robin Meta learned Hyperlink Ensembles -WebKB Tagging Round Robin Hyperlink Ensembles
With Round Robin, Meta Learned Hyperlink Ensembles outperforms Hyperlink Ensembles with PredHeadings and OwnText, and in half of the cases with PredListHeadings.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=39.95%** **ρ=22.75%** π=39.56% ρ=20.52% 3006	**π=41.36%** **ρ=25.4%** π=39.63% ρ=22.72% 3016	π=28.21% ρ=16.25% **π=28.91%** **ρ=18.2%** 3007	**π=40.69%** **ρ=19.96%** π=23.98% ρ=16.18% 3016	**π=41.21%** **ρ=24.29%** π=39.82% ρ=22.76% 3011	**π=27.57%** **ρ=28.83%** π=18.88% ρ=15.4% 8276
Pred LinkTags	**π=41.36%** **ρ=25.4%** π=39.63% ρ=22.72% 3016	π=32.63% ρ=21.17% **π=41.23%** **ρ=29.74%** 2940	π=24.13% ρ=16.75% **π=34.83%** **ρ=25.3%** 2941	**π=33.49%** **ρ=21.57%** π=32.75% ρ=18.81% 3001	π=32.56% ρ=21.88% **π=36.25%** **ρ=22.26%** 2954	π=33.98% ρ=35.3% **π=39.87%** **ρ=21.47%** 8276
PredList Headings	π=28.21% ρ=16.25% **π=28.91%** **ρ=18.2%** 3007	π=24.13% ρ=16.75% **π=34.83%** **ρ=25.3%** 2941	**π=32.32%** **ρ=21.21%** π=24.44% ρ=17.22% 1644	π=24.87% ρ=19.28% **π=30.31%** **ρ=18.64%** 2832	**π=28.3%** **ρ=16.48%** π=24.18% ρ=15.73% 2402	**π=31.69%** **ρ=33.89%** π=13.71% ρ=14.99% 8276
Pred Headings	**π=40.69%** **ρ=19.96%** π=23.98% ρ=16.18% 3016	**π=33.49%** **ρ=21.57%** π=32.75% ρ=18.81% 3001	π=24.87% ρ=19.28% **π=30.31%** **ρ=18.64%** 2832	**π=24.67%** **ρ=18.53%** π=22.8% ρ=16.05% 2828	π=26.05% ρ=19.31% **π=29%** **ρ=17.17%** 2911	**π=28.29%** **ρ=25.19%** π=14.97% ρ=14.5% 8276
PredLink Paragraph	**π=41.21%** **ρ=24.29%** π=39.82% ρ=22.76% 3011	π=32.56% ρ=21.88% **π=36.25%** **ρ=22.26%** 2954	**π=28.3%** **ρ=16.48%** π=24.18% ρ=15.73% 2402	π=26.05% ρ=19.31% **π=29%** **ρ=17.17%** 2911	π=28.25% ρ=17.15% **π=28.86%** **ρ=19.2%** 1143	**π=29.45%** **ρ=30.13%** π=26.25% ρ=17.52% 8276
Own Text	**π=27.57%** **ρ=28.83%** π=18.88% ρ=15.4% 8276	π=33.98% ρ=35.3% **π=39.87%** **ρ=21.47%** 8276	**π=31.69%** **ρ=33.89%** π=13.71% ρ=14.99% 8276	**π=28.29%** **ρ=25.19%** π=14.97% ρ=14.5% 8276	**π=29.45%** **ρ=30.13%** π=26.25% ρ=17.52% 8276	π=- ρ=- π=- ρ=- 8276

Table B.16: -WebKB Merging Round Robin Hyperlink Ensembles
With Merging and Round Robin, Meta Learned Hyperlink Ensembles outperforms Hyperlink Ensembles in the majority of the cases.

B.3 One Against All and Round Robin

Green if for One Against All and blue for Round Robin

B.3.1 Alleskar

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=84.65%** **ρ=67.3%** π=81.84% ρ=79.67% 3664	**π=84.89%** **ρ=65.67%** π=81.55% ρ=79.4% 3678	**π=84.87%** **ρ=67.31%** π=81.52% ρ=79.61% 3665	**π=84.15%** **ρ=63.8%** π=80.04% ρ=77.95% 3665	**π=82.72%** **ρ=58.88%** π=77.36% ρ=73.24% 3667	**π=82.58%** **ρ=58.44%** π=77.68% ρ=75.15% 3898
Pred LinkTags	**π=84.89%** **ρ=65.67%** π=81.55% ρ=79.4% 3678	**π=80%** **ρ=43.48%** π=64.12% ρ=57.35% 3653	**π=80.01%** **ρ=42.15%** π=63.87% ρ=57.07% 3653	**π=76.68%** **ρ=38.5%** π=59.22% ρ=53.72% 3653	**π=76.44%** **ρ=36.19%** π=59.38% ρ=52.68% 3655	**π=75.75%** **ρ=37.1%** π=63.11% ρ=56.63% 3898
PredList Headings	**π=84.87%** **ρ=67.31%** π=81.52% ρ=79.61% 3665	**π=80.01%** **ρ=42.15%** π=63.87% ρ=57.07% 3653	**π=70.18%** **ρ=26.66%** π=48.34% ρ=39.18% 1870	**π=71.83%** **ρ=28.78%** π=54.43% ρ=42.67% 2744	**π=79.66%** **ρ=26.77%** π=57.78% ρ=42.88% 3013	**π=72.36%** **ρ=33.82%** π=60.43% ρ=54.03% 3864
Pred Headings	**π=84.15%** **ρ=63.8%** π=80.04% ρ=77.95% 3665	**π=76.68%** **ρ=38.5%** π=59.22% ρ=53.72% 3653	**π=71.83%** **ρ=28.78%** π=54.43% ρ=42.67% 2744	**π=71.8%** **ρ=29.33%** π=55.42% ρ=44.09% 2672	**π=70.09%** **ρ=26.62%** π=54.6% ρ=40.5% 3103	**π=72.34%** **ρ=35.11%** π=61% ρ=54.56% 3879
PredLink Paragraph	**π=82.72%** **ρ=58.88%** π=77.36% ρ=73.24% 3667	**π=76.44%** **ρ=36.19%** π=59.38% ρ=52.68% 3655	**π=79.66%** **ρ=26.77%** π=57.78% ρ=42.88% 3013	**π=70.09%** **ρ=26.62%** π=54.6% ρ=40.5% 3103	**π=79.15%** **ρ=34.3%** π=64.88% ρ=51.51% 2715	**π=72.51%** **ρ=34.87%** π=60.88% ρ=55.4% 3882
Own Text	**π=82.58%** **ρ=58.44%** π=77.68% ρ=75.15% 3898	**π=75.75%** **ρ=37.1%** π=63.11% ρ=56.63% 3898	**π=72.36%** **ρ=33.82%** π=60.43% ρ=54.03% 3864	**π=72.34%** **ρ=35.11%** π=61% ρ=54.56% 3879	**π=72.51%** **ρ=34.87%** π=60.88% ρ=55.4% 3882	**π=71.67%** **ρ=32.17%** π=56.47% ρ=49.71% 3831

Table B.17: Alleskar Tagging One Against All Meta Predecessor -Alleskar Tagging Round Robin Meta Predecessor
Round Robin is outperformed by One against all.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=84.65%** **ρ=67.3%** π=81.84% ρ=79.67% 3664	**π=84.82%** **ρ=65.67%** π=81.94% ρ=79.62% 3678	**π=85.05%** **ρ=68.28%** π=81.4% ρ=79.5% 3665	**π=84.15%** **ρ=64.57%** π=79.76% ρ=77.84% 3665	**π=83.5%** **ρ=62.15%** π=79.25% ρ=76.33% 3667	**π=83.53%** **ρ=63.99%** π=79.58% ρ=77.93% 3898
Pred LinkTags	**π=84.82%** **ρ=65.67%** π=81.94% ρ=79.62% 3678	**π=80%** **ρ=43.48%** π=64.12% ρ=57.35% 3653	**π=79.71%** **ρ=43.63%** π=65.31% ρ=58.89% 3653	**π=77.74%** **ρ=40.17%** π=61.02% ρ=55.21% 3653	**π=78.43%** **ρ=39.77%** π=64.68% ρ=56.27% 3655	**π=79.14%** **ρ=41.05%** π=67.81% ρ=60.71% 3898
PredList Headings	**π=85.05%** **ρ=68.28%** π=81.4% ρ=79.5% 3665	**π=79.71%** **ρ=43.63%** π=65.31% ρ=58.89% 3653	**π=70.18%** **ρ=26.66%** π=48.34% ρ=39.18% 1870	**π=74.94%** **ρ=29.4%** π=55.1% ρ=43.64% 2744	**π=80.3%** **ρ=27.96%** π=60.3% ρ=44.85% 3013	**π=73.39%** **ρ=35.21%** π=60.5% ρ=54.57% 3864
Pred Headings	**π=84.15%** **ρ=64.57%** π=79.76% ρ=77.84% 3665	**π=77.74%** **ρ=40.17%** π=61.02% ρ=55.21% 3653	**π=74.94%** **ρ=29.4%** π=55.1% ρ=43.64% 2744	**π=71.8%** **ρ=29.33%** π=55.42% ρ=44.09% 2672	**π=74.2%** **ρ=29.17%** π=58.84% ρ=47.8% 3103	**π=73.88%** **ρ=36.11%** π=62.23% ρ=56.04% 3879
PredLink Paragraph	**π=83.5%** **ρ=62.15%** π=79.25% ρ=76.33% 3667	**π=78.43%** **ρ=39.77%** π=64.68% ρ=56.27% 3655	**π=80.3%** **ρ=27.96%** π=60.3% ρ=44.85% 3013	**π=74.2%** **ρ=29.17%** π=58.84% ρ=47.8% 3103	**π=79.15%** **ρ=34.3%** π=64.88% ρ=51.51% 2715	**π=75.58%** **ρ=38.68%** π=63.93% ρ=59.08% 3882
Own Text	**π=83.53%** **ρ=63.99%** π=79.58% ρ=77.93% 3898	**π=79.14%** **ρ=41.05%** π=67.81% ρ=60.71% 3898	**π=73.39%** **ρ=35.21%** π=60.5% ρ=54.57% 3864	**π=73.88%** **ρ=36.11%** π=62.23% ρ=56.04% 3879	**π=75.58%** **ρ=38.68%** π=63.93% ρ=59.08% 3882	**π=71.67%** **ρ=32.17%** π=56.47% ρ=49.71% 3831

Table B.18: Allesklar Merging One Against All Meta Predecessor -Allesklar Merging Round Robin Meta Predecessor
Round Robin is outperformed by One against all.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=82.97%** **ρ=46.92%** π=77.83% ρ=72.85% 3664	**π=83.41%** **ρ=47.31%** π=77.59% ρ=73.19% 3678	**π=83.06%** **ρ=47.07%** π=77.32% ρ=72.06% 3665	**π=82.89%** **ρ=38.34%** π=75.58% ρ=67.67% 3665	**π=83.25%** **ρ=45.75%** π=76.66% ρ=70.12% 3667	**π=81.88%** **ρ=39.39%** π=73.67% ρ=65.76% 3898
Pred LinkTags	**π=83.41%** **ρ=47.31%** π=77.59% ρ=73.19% 3678	**π=75.82%** **ρ=37.58%** π=59.15% ρ=48.9% 3653	**π=76.67%** **ρ=35.77%** π=56.62% ρ=47.74% 3653	**π=72.03%** **ρ=29.86%** π=56.91% ρ=40.13% 3653	**π=75.75%** **ρ=31.4%** π=55.31% ρ=44.83% 3655	**π=73.92%** **ρ=30.45%** π=55% ρ=44.37% 3898
PredList Headings	**π=83.06%** **ρ=47.07%** π=77.32% ρ=72.06% 3665	**π=76.67%** **ρ=35.77%** π=56.62% ρ=47.74% 3653	**π=68.19%** **ρ=28.54%** π=47.69% ρ=33.97% 1870	**π=67.81%** **ρ=28.37%** π=56.71% ρ=37% 2744	**π=77.62%** **ρ=27.61%** π=56.28% ρ=37.08% 3013	**π=71.62%** **ρ=27.81%** π=54.29% ρ=42.65% 3864
Pred Headings	**π=82.89%** **ρ=38.34%** π=75.58% ρ=67.67% 3665	**π=72.03%** **ρ=29.86%** π=56.91% ρ=40.13% 3653	**π=67.81%** **ρ=28.37%** π=56.71% ρ=37% 2744	**π=66.41%** **ρ=29.12%** π=59.11% ρ=37.65% 2672	**π=70.52%** **ρ=26.91%** π=61.39% ρ=38.76% 3103	**π=77.91%** **ρ=25.34%** π=56.34% ρ=38.08% 3879
PredLink Paragraph	**π=83.25%** **ρ=45.75%** π=76.66% ρ=70.12% 3667	**π=75.75%** **ρ=31.4%** π=55.31% ρ=44.83% 3655	**π=77.62%** **ρ=27.61%** π=56.28% ρ=37.08% 3013	**π=70.52%** **ρ=26.91%** π=61.39% ρ=38.76% 3103	**π=74.94%** **ρ=30.62%** π=63.3% ρ=41.72% 2715	**π=74.32%** **ρ=28.61%** π=59.7% ρ=42.89% 3882
Own Text	**π=81.88%** **ρ=39.39%** π=73.67% ρ=65.76% 3898	**π=73.92%** **ρ=30.45%** π=55% ρ=44.37% 3898	**π=71.62%** **ρ=27.81%** π=54.29% ρ=42.65% 3864	**π=77.91%** **ρ=25.34%** π=56.34% ρ=38.08% 3879	**π=74.32%** **ρ=28.61%** π=59.7% ρ=42.89% 3882	π=- ρ=- π=- ρ=- 3831

Table B.19: Allesklar Tagging One Against All Hyperlink Ensembles -Allesklar Tagging Round Robin Hyperlink Ensembles
Round Robin is outperformed by One against all.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=82.97%** **ρ=46.92%** π=77.83% ρ=72.85% 3664	**π=84.23%** **ρ=51.09%** π=78.87% ρ=74.84% 3678	**π=83.08%** **ρ=46.96%** π=77.76% ρ=72.58% 3665	**π=83.09%** **ρ=38.21%** π=76.14% ρ=67.9% 3665	**π=83.43%** **ρ=48.43%** π=78.15% ρ=72.95% 3667	**π=83.24%** **ρ=43.5%** π=78.69% ρ=73.23% 3898
Pred LinkTags	**π=84.23%** **ρ=51.09%** π=78.87% ρ=74.84% 3678	**π=75.82%** **ρ=37.58%** π=59.15% ρ=48.9% 3653	**π=77.05%** **ρ=35.79%** π=57.45% ρ=48.66% 3653	**π=76.87%** **ρ=31.86%** π=57.72% ρ=41.71% 3653	**π=77.71%** **ρ=35.36%** π=61.89% ρ=52.07% 3655	**π=76.46%** **ρ=33.01%** π=59.24% ρ=48.26% 3898
PredList Headings	**π=83.08%** **ρ=46.96%** π=77.76% ρ=72.58% 3665	**π=77.05%** **ρ=35.79%** π=57.45% ρ=48.66% 3653	**π=68.19%** **ρ=28.54%** π=47.69% ρ=33.97% 1870	**π=66.75%** **ρ=27.95%** π=57.59% ρ=37.26% 2744	**π=78.48%** **ρ=28.62%** π=56.74% ρ=38.36% 3013	**π=75.07%** **ρ=26.79%** π=59.29% ρ=34.18% 3864
Pred Headings	**π=83.09%** **ρ=38.21%** π=76.14% ρ=67.9% 3665	**π=76.87%** **ρ=31.86%** π=57.72% ρ=41.71% 3653	**π=66.75%** **ρ=27.95%** π=57.59% ρ=37.26% 2744	**π=66.41%** **ρ=29.12%** π=59.11% ρ=37.65% 2672	**π=71.88%** **ρ=25.87%** π=61.86% ρ=38.15% 3103	**π=72.64%** **ρ=25.14%** π=60.5% ρ=35.27% 3879
PredLink Paragraph	**π=83.43%** **ρ=48.43%** π=78.15% ρ=72.95% 3667	**π=77.71%** **ρ=35.36%** π=61.89% ρ=52.07% 3655	**π=78.48%** **ρ=28.62%** π=56.74% ρ=38.36% 3013	**π=71.88%** **ρ=25.87%** π=61.86% ρ=38.15% 3103	**π=74.94%** **ρ=30.62%** π=63.3% ρ=41.72% 2715	**π=78.35%** **ρ=30.39%** π=65.51% ρ=45.06% 3882
Own Text	**π=83.24%** **ρ=43.5%** π=78.69% ρ=73.23% 3898	**π=76.46%** **ρ=33.01%** π=59.24% ρ=48.26% 3898	**π=75.07%** **ρ=26.79%** π=59.29% ρ=34.18% 3864	**π=72.64%** **ρ=25.14%** π=60.5% ρ=35.27% 3879	**π=78.35%** **ρ=30.39%** π=65.51% ρ=45.06% 3882	π=- ρ=- π=- ρ=- 3831

Table B.20: Allesklar Merging One Against All Hyperlink Ensembles -Allesklar Merging Round Robin Hyperlink Ensembles
Round Robin is outperformed by One against all.

B.3.2 WebKB

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=**41.07%** ρ=**17.94%** π=40.08% ρ=19.46% 3006	π=**56.66%** ρ=**20.35%** π=49.43% ρ=21.95% 3016	π=30.13% ρ=16.91% π=**34.29%** ρ=**17.96%** 3007	π=36.49% ρ=17.36% π=**36.96%** ρ=**18.13%** 3016	π=35.51% ρ=19.08% π=**39.29%** ρ=**19.89%** 3011	π=**44.27%** ρ=**24.31%** π=42.27% ρ=28.96% 8276
Pred LinkTags	π=**56.66%** ρ=**20.35%** π=49.43% ρ=21.95% 3016	π=**35.54%** ρ=**21.35%** π=34.16% ρ=21.86% 2940	π=34.02% ρ=19% π=**35.53%** ρ=**19.86%** 2941	π=**29.23%** ρ=**17.36%** π=27.21% ρ=17.85% 3001	π=**30.53%** ρ=**19.87%** π=29.66% ρ=19.89% 2954	π=**43.23%** ρ=**24.44%** π=42.76% ρ=29.18% 8276
PredList Headings	π=30.13% ρ=16.91% π=**34.29%** ρ=**17.96%** 3007	π=34.02% ρ=19% π=**35.53%** ρ=**19.86%** 2941	π=17.38% ρ=14.89% π=**30.7%** ρ=**19.42%** 1644	π=**27.86%** ρ=**17.3%** π=25.11% ρ=17.58% 2832	π=**26.14%** ρ=**16.65%** π=25.7% ρ=17.18% 2402	π=**43.71%** ρ=**24.02%** π=41.85% ρ=28.53% 8276
Pred Headings	π=36.49% ρ=17.36% π=**36.96%** ρ=**18.13%** 3016	π=**29.23%** ρ=**17.36%** π=27.21% ρ=17.85% 3001	π=**27.86%** ρ=**17.3%** π=25.11% ρ=17.58% 2832	π=**28.35%** ρ=**17.37%** π=25.62% ρ=16.64% 2828	π=**26.13%** ρ=**16.84%** π=24.5% ρ=17.1% 2911	π=**43.96%** ρ=**23.65%** π=41.72% ρ=28.36% 8276
PredLink Paragraph	π=35.51% ρ=19.08% π=**39.29%** ρ=**19.89%** 3011	π=**30.53%** ρ=**19.87%** π=29.66% ρ=19.89% 2954	π=**26.14%** ρ=**16.65%** π=25.7% ρ=17.18% 2402	π=**26.13%** ρ=**16.84%** π=24.5% ρ=17.1% 2911	π=**29.17%** ρ=**16.71%** π=27.79% ρ=16.86% 1143	π=**43.5%** ρ=**24.69%** π=41.51% ρ=28.86% 8276
Own Text	π=**44.27%** ρ=**24.31%** π=42.27% ρ=28.96% 8276	π=**43.23%** ρ=**24.44%** π=42.76% ρ=29.18% 8276	π=**43.71%** ρ=**24.02%** π=41.85% ρ=28.53% 8276	π=**43.96%** ρ=**23.65%** π=41.72% ρ=28.36% 8276	π=**43.5%** ρ=**24.69%** π=41.51% ρ=28.86% 8276	π=**45.37%** ρ=**24.71%** π=42% ρ=29.13% 8276

Table B.21: WebKB Tagging One Against All Meta Predecessor -WebKB Tagging Round Robin Meta Predecessor
Round Robin outperforms One Against All in half of the combinations including WordsAround or PredListHeadings.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	**π=41.07%** **ρ=17.94%** π=40.08% ρ=19.46% 3006	**π=44.4%** **ρ=21.05%** π=41.85% ρ=22.57% 3016	π=28.08% ρ=15.3% **π=30.79%** **ρ=16.08%** 3007	π=37.51% ρ=16.95% **π=38.14%** **ρ=17.37%** 3016	**π=42.74%** **ρ=19.43%** π=41.58% ρ=20.81% 3011	**π=40.45%** **ρ=21.79%** π=40.1% ρ=26.22% 8276
Pred LinkTags	**π=44.4%** **ρ=21.05%** π=41.85% ρ=22.57% 3016	**π=35.54%** **ρ=21.35%** π=34.16% ρ=21.86% 2940	π=23.48% ρ=16.11% **π=24.26%** **ρ=16.6%** 2941	**π=32.96%** **ρ=16.34%** π=30.5% ρ=17.6% 3001	π=30.5% ρ=20.24% **π=31.18%** **ρ=20.56%** 2954	**π=43.01%** **ρ=23.82%** π=41.63% ρ=27.88% 8276
PredList Headings	π=28.08% ρ=15.3% **π=30.79%** **ρ=16.08%** 3007	π=23.48% ρ=16.11% **π=24.26%** **ρ=16.6%** 2941	π=17.38% ρ=14.89% **π=30.7%** **ρ=19.42%** 1644	**π=30.41%** **ρ=17.71%** π=27.63% ρ=17.29% 2832	π=19.99% ρ=14.9% **π=28.76%** **ρ=16.03%** 2402	**π=42.29%** **ρ=23.29%** π=39.42% ρ=26.96% 8276
Pred Headings	π=37.51% ρ=16.95% **π=38.14%** **ρ=17.37%** 3016	**π=32.96%** **ρ=16.34%** π=30.5% ρ=17.6% 3001	**π=30.41%** **ρ=17.71%** π=27.63% ρ=17.29% 2832	**π=28.35%** **ρ=17.37%** π=25.62% ρ=16.64% 2828	π=26.55% ρ=16.73% **π=26.62%** **ρ=17.19%** 2911	**π=42.83%** **ρ=23.12%** π=40.76% ρ=27.28% 8276
PredLink Paragraph	**π=42.74%** **ρ=19.43%** π=41.58% ρ=20.81% 3011	π=30.5% ρ=20.24% **π=31.18%** **ρ=20.56%** 2954	π=19.99% ρ=14.9% **π=28.76%** **ρ=16.03%** 2402	π=26.55% ρ=16.73% **π=26.62%** **ρ=17.19%** 2911	**π=29.17%** **ρ=16.71%** π=27.79% ρ=16.86% 1143	**π=42.45%** **ρ=23.76%** π=41.02% ρ=28.37% 8276
Own Text	**π=40.45%** **ρ=21.79%** π=40.1% ρ=26.22% 8276	**π=43.01%** **ρ=23.82%** π=41.63% ρ=27.88% 8276	**π=42.29%** **ρ=23.29%** π=39.42% ρ=26.96% 8276	**π=42.83%** **ρ=23.12%** π=40.76% ρ=27.28% 8276	**π=42.45%** **ρ=23.76%** π=41.02% ρ=28.37% 8276	**π=45.37%** **ρ=24.71%** π=42% ρ=29.13% 8276

Table B.22: WebKB Merging One Against All Meta Predecessor -WebKB Merging Round Robin Meta Predecessor

Round Robin outperforms One Against All in half of the combinations including PredLinkParagraph or PredListHeadings.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=36.85% ρ=18.48% **π=39.56%** **ρ=20.52%** 3006	**π=52.46%** **ρ=20.98%** π=51.6% ρ=24.26% 3016	**π=33.14%** **ρ=18.3%** π=32.95% ρ=19.99% 3007	π=26.85% ρ=16.48% **π=27.39%** **ρ=17.03%** 3016	π=39.79% ρ=19.01% **π=40.38%** **ρ=22.64%** 3011	**π=40.76%** **ρ=22.45%** π=37.23% ρ=25.37% 8276
Pred LinkTags	**π=52.46%** **ρ=20.98%** π=51.6% ρ=24.26% 3016	**π=41.99%** **ρ=27.35%** π=41.23% ρ=29.74% 2940	**π=47.3%** **ρ=24.95%** π=39.01% ρ=26.41% 2941	**π=33.09%** **ρ=18.49%** π=30.63% ρ=19.05% 3001	**π=34.51%** **ρ=19.58%** π=33.85% ρ=22.74% 2954	**π=44.31%** **ρ=22.63%** π=39.78% ρ=24.96% 8276
PredList Headings	**π=33.14%** **ρ=18.3%** π=32.95% ρ=19.99% 3007	**π=47.3%** **ρ=24.95%** π=39.01% ρ=26.41% 2941	π=24.39% ρ=15.9% **π=24.44%** **ρ=17.22%** 1644	**π=30.09%** **ρ=18.91%** π=26.04% ρ=17.45% 2832	**π=27.82%** **ρ=16.2%** π=26.04% ρ=16.54% 2402	**π=40.46%** **ρ=23.28%** π=38.55% ρ=26.67% 8276
Pred Headings	π=26.85% ρ=16.48% **π=27.39%** **ρ=17.03%** 3016	**π=33.09%** **ρ=18.49%** π=30.63% ρ=19.05% 3001	**π=30.09%** **ρ=18.91%** π=26.04% ρ=17.45% 2832	π=20.32% ρ=15.7% **π=22.8%** **ρ=16.05%** 2828	**π=26.82%** **ρ=16.89%** π=24.46% ρ=16.9% 2911	**π=36.67%** **ρ=19.26%** π=35.23% ρ=21.44% 8276
PredLink Paragraph	π=39.79% ρ=19.01% **π=40.38%** **ρ=22.64%** 3011	**π=34.51%** **ρ=19.58%** π=33.85% ρ=22.74% 2954	**π=27.82%** **ρ=16.2%** π=26.04% ρ=16.54% 2402	**π=26.82%** **ρ=16.89%** π=24.46% ρ=16.9% 2911	**π=29.23%** **ρ=18.03%** π=28.86% ρ=19.2% 1143	π=41.65% ρ=24.02% **π=42.39%** **ρ=28.29%** 8276
Own Text	**π=40.76%** **ρ=22.45%** π=37.23% ρ=25.37% 8276	**π=44.31%** **ρ=22.63%** π=39.78% ρ=24.96% 8276	**π=40.46%** **ρ=23.28%** π=38.55% ρ=26.67% 8276	**π=36.67%** **ρ=19.26%** π=35.23% ρ=21.44% 8276	π=41.65% ρ=24.02% **π=42.39%** **ρ=28.29%** 8276	π=- ρ=- π=- ρ=- 8276

Table B.23: WebKB Tagging One Against All Hyperlink Ensembles -WebKB Tagging Round Robin Hyperlink Ensembles
Round Robin outperforms One Against All in half of the combinations including WordsAround.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=36.85% ρ=18.48% **π=39.56%** **ρ=20.52%** 3006	π=39.09% ρ=19.59% **π=39.63%** **ρ=22.72%** 3016	**π=29.91%** **ρ=15.19%** π=28.91% ρ=18.2% 3007	π=16.02% ρ=15.45% **π=23.98%** **ρ=16.18%** 3016	**π=40.71%** **ρ=19.05%** π=39.82% ρ=22.76% 3011	π=17.88% ρ=14.77% **π=18.88%** **ρ=15.4%** 8276
Pred LinkTags	π=39.09% ρ=19.59% **π=39.63%** **ρ=22.72%** 3016	**π=41.99%** **ρ=27.35%** π=41.23% ρ=29.74% 2940	**π=36.74%** **ρ=23.47%** π=34.83% ρ=25.3% 2941	**π=36.96%** **ρ=17.8%** π=32.75% ρ=18.81% 3001	π=34.4% ρ=20.16% **π=36.25%** **ρ=22.26%** 2954	π=39.31% ρ=21.09% **π=39.87%** **ρ=21.47%** 8276
PredList Headings	**π=29.91%** **ρ=15.19%** π=28.91% ρ=18.2% 3007	**π=36.74%** **ρ=23.47%** π=34.83% ρ=25.3% 2941	π=24.39% ρ=15.9% **π=24.44%** **ρ=17.22%** 1644	π=25.56% ρ=16.95% **π=30.31%** **ρ=18.64%** 2832	π=23.98% ρ=15.39% **π=24.18%** **ρ=15.73%** 2402	**π=14.66%** **ρ=15%** π=13.71% ρ=14.99% 8276
Pred Headings	π=16.02% ρ=15.45% **π=23.98%** **ρ=16.18%** 3016	**π=36.96%** **ρ=17.8%** π=32.75% ρ=18.81% 3001	π=25.56% ρ=16.95% **π=30.31%** **ρ=18.64%** 2832	π=20.32% ρ=15.7% **π=22.8%** **ρ=16.05%** 2828	π=20.93% ρ=15.84% **π=29%** **ρ=17.17%** 2911	**π=17.16%** **ρ=14.56%** π=14.97% ρ=14.5% 8276
PredLink Paragraph	**π=40.71%** **ρ=19.05%** π=39.82% ρ=22.76% 3011	π=34.4% ρ=20.16% **π=36.25%** **ρ=22.26%** 2954	π=23.98% ρ=15.39% **π=24.18%** **ρ=15.73%** 2402	π=20.93% ρ=15.84% **π=29%** **ρ=17.17%** 2911	**π=29.23%** **ρ=18.03%** π=28.86% ρ=19.2% 1143	π=25.35% ρ=16.53% **π=26.25%** **ρ=17.52%** 8276
Own Text	π=17.88% ρ=14.77% **π=18.88%** **ρ=15.4%** 8276	π=39.31% ρ=21.09% **π=39.87%** **ρ=21.47%** 8276	**π=14.66%** **ρ=15%** π=13.71% ρ=14.99% 8276	**π=17.16%** **ρ=14.56%** π=14.97% ρ=14.5% 8276	π=25.35% ρ=16.53% **π=26.25%** **ρ=17.52%** 8276	π=- ρ=- π=- ρ=- 8276

Table B.24: WebKB Merging One Against All Hyperlink Ensembles -WebKB Merging Round Robin Hyperlink Ensembles

One Against All outperforms Round Robin in a big majority of cases. But the average precision reached by both of those methods is quite bad.

B.4 Tagging and Merging

Green if for Tagging and blue for Merging

B.4.1 Alleskar

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=84.65% ρ=67.3% π=84.65% ρ=67.3% 3664	**π=84.89%** **ρ=65.67%** π=84.82% ρ=65.67% 3678	π=84.87% ρ=67.31% **π=85.05%** **ρ=68.28%** 3665	π=84.15% ρ=63.8% **π=84.15%** **ρ=64.57%** 3665	π=82.72% ρ=58.88% **π=83.5%** **ρ=62.15%** 3667	π=82.58% ρ=58.44% **π=83.53%** **ρ=63.99%** 3898
Pred LinkTags	**π=84.89%** **ρ=65.67%** π=84.82% ρ=65.67% 3678	π=80% ρ=43.48% π=80% ρ=43.48% 3653	**π=80.01%** **ρ=42.15%** π=79.71% ρ=43.63% 3653	π=76.68% ρ=38.5% **π=77.74%** **ρ=40.17%** 3653	π=76.44% ρ=36.19% **π=78.43%** **ρ=39.77%** 3655	π=75.75% ρ=37.1% **π=79.14%** **ρ=41.05%** 3898
PredList Headings	π=84.87% ρ=67.31% **π=85.05%** **ρ=68.28%** 3665	**π=80.01%** **ρ=42.15%** π=79.71% ρ=43.63% 3653	π=70.18% ρ=26.66% π=70.18% ρ=26.66% 1870	π=71.83% ρ=28.78% **π=74.94%** **ρ=29.4%** 2744	π=79.66% ρ=26.77% **π=80.3%** **ρ=27.96%** 3013	π=72.36% ρ=33.82% **π=73.39%** **ρ=35.21%** 3864
Pred Headings	π=84.15% ρ=63.8% **π=84.15%** **ρ=64.57%** 3665	π=76.68% ρ=38.5% **π=77.74%** **ρ=40.17%** 3653	π=71.83% ρ=28.78% **π=74.94%** **ρ=29.4%** 2744	π=71.8% ρ=29.33% π=71.8% ρ=29.33% 2672	π=70.09% ρ=26.62% **π=74.2%** **ρ=29.17%** 3103	π=72.34% ρ=35.11% **π=73.88%** **ρ=36.11%** 3879
PredLink Paragraph	π=82.72% ρ=58.88% **π=83.5%** **ρ=62.15%** 3667	π=76.44% ρ=36.19% **π=78.43%** **ρ=39.77%** 3655	π=79.66% ρ=26.77% **π=80.3%** **ρ=27.96%** 3013	π=70.09% ρ=26.62% **π=74.2%** **ρ=29.17%** 3103	π=79.15% ρ=34.3% π=79.15% ρ=34.3% 2715	π=72.51% ρ=34.87% **π=75.58%** **ρ=38.68%** 3882
Own Text	π=82.58% ρ=58.44% **π=83.53%** **ρ=63.99%** 3898	π=75.75% ρ=37.1% **π=79.14%** **ρ=41.05%** 3898	π=72.36% ρ=33.82% **π=73.39%** **ρ=35.21%** 3864	π=72.34% ρ=35.11% **π=73.88%** **ρ=36.11%** 3879	π=72.51% ρ=34.87% **π=75.58%** **ρ=38.68%** 3882	π=71.67% ρ=32.17% π=71.67% ρ=32.17% 3831

Table B.25: Alleskar Tagging One Against All Meta Predecessor -Alleskar Merging One Against All Meta Predecessor
Merging outperforms Tagging in almost all the cases.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=82.97% ρ=46.92% π=82.97% ρ=46.92% 3664	π=83.41% ρ=47.31% **π=84.23%** **ρ=51.09%** 3678	π=83.06% ρ=47.07% **π=83.08%** **ρ=46.96%** 3665	π=82.89% ρ=38.34% **π=83.09%** **ρ=38.21%** 3665	π=83.25% ρ=45.75% **π=83.43%** **ρ=48.43%** 3667	π=81.88% ρ=39.39% **π=83.24%** **ρ=43.5%** 3898
Pred LinkTags	π=83.41% ρ=47.31% **π=84.23%** **ρ=51.09%** 3678	π=75.82% ρ=37.58% π=75.82% ρ=37.58% 3653	π=76.67% ρ=35.77% **π=77.05%** **ρ=35.79%** 3653	π=72.03% ρ=29.86% **π=76.87%** **ρ=31.86%** 3653	π=75.75% ρ=31.4% **π=77.71%** **ρ=35.36%** 3655	π=73.92% ρ=30.45% **π=76.46%** **ρ=33.01%** 3898
PredList Headings	π=83.06% ρ=47.07% **π=83.08%** **ρ=46.96%** 3665	π=76.67% ρ=35.77% **π=77.05%** **ρ=35.79%** 3653	π=68.19% ρ=28.54% π=68.19% ρ=28.54% 1870	**π=67.81%** **ρ=28.37%** π=66.75% ρ=27.95% 2744	π=77.62% ρ=27.61% **π=78.48%** **ρ=28.62%** 3013	π=71.62% ρ=27.81% **π=75.07%** **ρ=26.79%** 3864
Pred Headings	π=82.89% ρ=38.34% **π=83.09%** **ρ=38.21%** 3665	π=72.03% ρ=29.86% **π=76.87%** **ρ=31.86%** 3653	**π=67.81%** **ρ=28.37%** π=66.75% ρ=27.95% 2744	π=66.41% ρ=29.12% π=66.41% ρ=29.12% 2672	π=70.52% ρ=26.91% **π=71.88%** **ρ=25.87%** 3103	**π=77.91%** **ρ=25.34%** π=72.64% ρ=25.14% 3879
PredLink Paragraph	π=83.25% ρ=45.75% **π=83.43%** **ρ=48.43%** 3667	π=75.75% ρ=31.4% **π=77.71%** **ρ=35.36%** 3655	π=77.62% ρ=27.61% **π=78.48%** **ρ=28.62%** 3013	π=70.52% ρ=26.91% **π=71.88%** **ρ=25.87%** 3103	π=74.94% ρ=30.62% π=74.94% ρ=30.62% 2715	π=74.32% ρ=28.61% **π=78.35%** **ρ=30.39%** 3882
Own Text	π=81.88% ρ=39.39% **π=83.24%** **ρ=43.5%** 3898	π=73.92% ρ=30.45% **π=76.46%** **ρ=33.01%** 3898	π=71.62% ρ=27.81% **π=75.07%** **ρ=26.79%** 3864	**π=77.91%** **ρ=25.34%** π=72.64% ρ=25.14% 3879	π=74.32% ρ=28.61% **π=78.35%** **ρ=30.39%** 3882	π=- ρ=- π=- ρ=- 3831

Table B.26: Alleskla r Tagging One Against All Hyperlink Ensembles -Alleskla r Merging One Against All Hyperlink Ensembles
Merging outperforms Tagging in almost all the cases.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=81.84% ρ=79.67% π=81.84% ρ=79.67% 3664	π=81.55% ρ=79.4% **π=81.94%** **ρ=79.62%** 3678	**π=81.52%** **ρ=79.61%** π=81.4% ρ=79.5% 3665	**π=80.04%** **ρ=77.95%** π=79.76% ρ=77.84% 3665	π=77.36% ρ=73.24% **π=79.25%** **ρ=76.33%** 3667	π=77.68% ρ=75.15% **π=79.58%** **ρ=77.93%** 3898
Pred LinkTags	π=81.55% ρ=79.4% **π=81.94%** **ρ=79.62%** 3678	π=64.12% ρ=57.35% π=64.12% ρ=57.35% 3653	π=63.87% ρ=57.07% **π=65.31%** **ρ=58.89%** 3653	π=59.22% ρ=53.72% **π=61.02%** **ρ=55.21%** 3653	π=59.38% ρ=52.68% **π=64.68%** **ρ=56.27%** 3655	π=63.11% ρ=56.63% **π=67.81%** **ρ=60.71%** 3898
PredList Headings	**π=81.52%** **ρ=79.61%** π=81.4% ρ=79.5% 3665	π=63.87% ρ=57.07% **π=65.31%** **ρ=58.89%** 3653	π=48.34% ρ=39.18% π=48.34% ρ=39.18% 1870	π=54.43% ρ=42.67% **π=55.1%** **ρ=43.64%** 2744	π=57.78% ρ=42.88% **π=60.3%** **ρ=44.85%** 3013	π=60.43% ρ=54.03% **π=60.5%** **ρ=54.57%** 3864
Pred Headings	**π=80.04%** **ρ=77.95%** π=79.76% ρ=77.84% 3665	π=59.22% ρ=53.72% **π=61.02%** **ρ=55.21%** 3653	π=54.43% ρ=42.67% **π=55.1%** **ρ=43.64%** 2744	π=55.42% ρ=44.09% π=55.42% ρ=44.09% 2672	π=54.6% ρ=40.5% **π=58.84%** **ρ=47.8%** 3103	π=61% ρ=54.56% **π=62.23%** **ρ=56.04%** 3879
PredLink Paragraph	π=77.36% ρ=73.24% **π=79.25%** **ρ=76.33%** 3667	π=59.38% ρ=52.68% **π=64.68%** **ρ=56.27%** 3655	π=57.78% ρ=42.88% **π=60.3%** **ρ=44.85%** 3013	π=54.6% ρ=40.5% **π=58.84%** **ρ=47.8%** 3103	π=64.88% ρ=51.51% π=64.88% ρ=51.51% 2715	π=60.88% ρ=55.4% **π=63.93%** **ρ=59.08%** 3882
Own Text	π=77.68% ρ=75.15% **π=79.58%** **ρ=77.93%** 3898	π=63.11% ρ=56.63% **π=67.81%** **ρ=60.71%** 3898	π=60.43% ρ=54.03% **π=60.5%** **ρ=54.57%** 3864	π=61% ρ=54.56% **π=62.23%** **ρ=56.04%** 3879	π=60.88% ρ=55.4% **π=63.93%** **ρ=59.08%** 3882	π=56.47% ρ=49.71% π=56.47% ρ=49.71% 3831

Table B.27: Alleksklar Tagging Round Robin Meta Predecessor -Alleskslar Merging Round Robin Meta Predecessor
Merging outperforms Tagging in almost all the cases.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=77.83% ρ=72.85% π=77.83% ρ=72.85% 3664	π=77.59% ρ=73.19% **π=78.87%** **ρ=74.84%** 3678	π=77.32% ρ=72.06% **π=77.76%** **ρ=72.58%** 3665	π=75.58% ρ=67.67% **π=76.14%** **ρ=67.9%** 3665	π=76.66% ρ=70.12% **π=78.15%** **ρ=72.95%** 3667	π=73.67% ρ=65.76% **π=78.69%** **ρ=73.23%** 3898
Pred LinkTags	π=77.59% ρ=73.19% **π=78.87%** **ρ=74.84%** 3678	π=59.15% ρ=48.9% π=59.15% ρ=48.9% 3653	π=56.62% ρ=47.74% **π=57.45%** **ρ=48.66%** 3653	π=56.91% ρ=40.13% **π=57.72%** **ρ=41.71%** 3653	π=55.31% ρ=44.83% **π=61.89%** **ρ=52.07%** 3655	π=55% ρ=44.37% **π=59.24%** **ρ=48.26%** 3898
PredList Headings	π=77.32% ρ=72.06% **π=77.76%** **ρ=72.58%** 3665	π=56.62% ρ=47.74% **π=57.45%** **ρ=48.66%** 3653	π=47.69% ρ=33.97% π=47.69% ρ=33.97% 1870	π=56.71% ρ=37% **π=57.59%** **ρ=37.26%** 2744	π=56.28% ρ=37.08% **π=56.74%** **ρ=38.36%** 3013	π=54.29% ρ=42.65% **π=59.29%** **ρ=34.18%** 3864
Pred Headings	π=75.58% ρ=67.67% **π=76.14%** **ρ=67.9%** 3665	π=56.91% ρ=40.13% **π=57.72%** **ρ=41.71%** 3653	π=56.71% ρ=37% **π=57.59%** **ρ=37.26%** 2744	π=59.11% ρ=37.65% π=59.11% ρ=37.65% 2672	π=61.39% ρ=38.76% **π=61.86%** **ρ=38.15%** 3103	π=56.34% ρ=38.08% **π=60.5%** **ρ=35.27%** 3879
PredLink Paragraph	π=76.66% ρ=70.12% **π=78.15%** **ρ=72.95%** 3667	π=55.31% ρ=44.83% **π=61.89%** **ρ=52.07%** 3655	π=56.28% ρ=37.08% **π=56.74%** **ρ=38.36%** 3013	π=61.39% ρ=38.76% **π=61.86%** **ρ=38.15%** 3103	π=63.3% ρ=41.72% π=63.3% ρ=41.72% 2715	π=59.7% ρ=42.89% **π=65.51%** **ρ=45.06%** 3882
Own Text	π=73.67% ρ=65.76% **π=78.69%** **ρ=73.23%** 3898	π=55% ρ=44.37% **π=59.24%** **ρ=48.26%** 3898	π=54.29% ρ=42.65% **π=59.29%** **ρ=34.18%** 3864	π=56.34% ρ=38.08% **π=60.5%** **ρ=35.27%** 3879	π=59.7% ρ=42.89% **π=65.51%** **ρ=45.06%** 3882	π=- ρ=- π=- ρ=- 3831

Table B.28: Allesklar Tagging Round Robin Hyperlink Ensembles -Allesklar Merging Round Robin Hyperlink Ensembles
Merging outperforms Tagging in all the cases.

B.4.2 WebKB

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=41.07% ρ=17.94% π=41.07% ρ=17.94% 3006	**π=56.66%** **ρ=20.35%** π=44.4% ρ=21.05% 3016	**π=30.13%** **ρ=16.91%** π=28.08% ρ=15.3% 3007	π=36.49% ρ=17.36% **π=37.51%** **ρ=16.95%** 3016	π=35.51% ρ=19.08% **π=42.74%** **ρ=19.43%** 3011	**π=44.27%** **ρ=24.31%** π=40.45% ρ=21.79% 8276
Pred LinkTags	**π=56.66%** **ρ=20.35%** π=44.4% ρ=21.05% 3016	π=35.54% ρ=21.35% π=35.54% ρ=21.35% 2940	**π=34.02%** **ρ=19%** π=23.48% ρ=16.11% 2941	π=29.23% ρ=17.36% **π=32.96%** **ρ=16.34%** 3001	**π=30.53%** **ρ=19.87%** π=30.5% ρ=20.24% 2954	**π=43.23%** **ρ=24.44%** π=43.01% ρ=23.82% 8276
PredList Headings	**π=30.13%** **ρ=16.91%** π=28.08% ρ=15.3% 3007	**π=34.02%** **ρ=19%** π=23.48% ρ=16.11% 2941	π=17.38% ρ=14.89% π=17.38% ρ=14.89% 1644	π=27.86% ρ=17.3% **π=30.41%** **ρ=17.71%** 2832	**π=26.14%** **ρ=16.65%** π=19.99% ρ=14.9% 2402	**π=43.71%** **ρ=24.02%** π=42.29% ρ=23.29% 8276
Pred Headings	π=36.49% ρ=17.36% **π=37.51%** **ρ=16.95%** 3016	π=29.23% ρ=17.36% **π=32.96%** **ρ=16.34%** 3001	π=27.86% ρ=17.3% **π=30.41%** **ρ=17.71%** 2832	π=28.35% ρ=17.37% π=28.35% ρ=17.37% 2828	π=26.13% ρ=16.84% **π=26.55%** **ρ=16.73%** 2911	**π=43.96%** **ρ=23.65%** π=42.83% ρ=23.12% 8276
PredLink Paragraph	π=35.51% ρ=19.08% **π=42.74%** **ρ=19.43%** 3011	**π=30.53%** **ρ=19.87%** π=30.5% ρ=20.24% 2954	**π=26.14%** **ρ=16.65%** π=19.99% ρ=14.9% 2402	π=26.13% ρ=16.84% **π=26.55%** **ρ=16.73%** 2911	π=29.17% ρ=16.71% π=29.17% ρ=16.71% 1143	**π=43.5%** **ρ=24.69%** π=42.45% ρ=23.76% 8276
Own Text	**π=44.27%** **ρ=24.31%** π=40.45% ρ=21.79% 8276	**π=43.23%** **ρ=24.44%** π=43.01% ρ=23.82% 8276	**π=43.71%** **ρ=24.02%** π=42.29% ρ=23.29% 8276	**π=43.96%** **ρ=23.65%** π=42.83% ρ=23.12% 8276	**π=43.5%** **ρ=24.69%** π=42.45% ρ=23.76% 8276	π=45.37% ρ=24.71% π=45.37% ρ=24.71% 8276

Table B.29: WebKB Tagging One Against All Meta Predecessor -WebKB Merging One Against All Meta Predecessor
The results are equally distributed: Each method between Merging and Tagging wins one-half of the matches.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=36.85% ρ=18.48% π=36.85% ρ=18.48% 3006	**π=52.46%** **ρ=20.98%** π=39.09% ρ=19.59% 3016	**π=33.14%** **ρ=18.3%** π=29.91% ρ=15.19% 3007	**π=26.85%** **ρ=16.48%** π=16.02% ρ=15.45% 3016	π=39.79% ρ=19.01% **π=40.71%** **ρ=19.05%** 3011	**π=40.76%** **ρ=22.45%** π=17.88% ρ=14.77% 8276
Pred LinkTags	**π=52.46%** **ρ=20.98%** π=39.09% ρ=19.59% 3016	π=41.99% ρ=27.35% π=41.99% ρ=27.35% 2940	**π=47.3%** **ρ=24.95%** π=36.74% ρ=23.47% 2941	π=33.09% ρ=18.49% **π=36.96%** **ρ=17.8%** 3001	**π=34.51%** **ρ=19.58%** π=34.4% ρ=20.16% 2954	**π=44.31%** **ρ=22.63%** π=39.31% ρ=21.09% 8276
PredList Headings	**π=33.14%** **ρ=18.3%** π=29.91% ρ=15.19% 3007	**π=47.3%** **ρ=24.95%** π=36.74% ρ=23.47% 2941	π=24.39% ρ=15.9% π=24.39% ρ=15.9% 1644	**π=30.09%** **ρ=18.91%** π=25.56% ρ=16.95% 2832	**π=27.82%** **ρ=16.2%** π=23.98% ρ=15.39% 2402	**π=40.46%** **ρ=23.28%** π=14.66% ρ=15% 8276
Pred Headings	**π=26.85%** **ρ=16.48%** π=16.02% ρ=15.45% 3016	π=33.09% ρ=18.49% **π=36.96%** **ρ=17.8%** 3001	**π=30.09%** **ρ=18.91%** π=25.56% ρ=16.95% 2832	π=20.32% ρ=15.7% π=20.32% ρ=15.7% 2828	**π=26.82%** **ρ=16.89%** π=20.93% ρ=15.84% 2911	**π=36.67%** **ρ=19.26%** π=17.16% ρ=14.56% 8276
PredLink Paragraph	π=39.79% ρ=19.01% **π=40.71%** **ρ=19.05%** 3011	**π=34.51%** **ρ=19.58%** π=34.4% ρ=20.16% 2954	**π=27.82%** **ρ=16.2%** π=23.98% ρ=15.39% 2402	**π=26.82%** **ρ=16.89%** π=20.93% ρ=15.84% 2911	π=29.23% ρ=18.03% π=29.23% ρ=18.03% 1143	**π=41.65%** **ρ=24.02%** π=25.35% ρ=16.53% 8276
Own Text	**π=40.76%** **ρ=22.45%** π=17.88% ρ=14.77% 8276	**π=44.31%** **ρ=22.63%** π=39.31% ρ=21.09% 8276	**π=40.46%** **ρ=23.28%** π=14.66% ρ=15% 8276	**π=36.67%** **ρ=19.26%** π=17.16% ρ=14.56% 8276	**π=41.65%** **ρ=24.02%** π=25.35% ρ=16.53% 8276	π=- ρ=- π=- ρ=- 8276

Table B.30: WebKB Tagging One Against All Hyperlink Ensembles -WebKB Merging One Against All Hyperlink Ensembles
Tagging outperforms Merging in the majority of the cases.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=40.08% ρ=19.46% π=40.08% ρ=19.46% 3006	**π=49.43%** **ρ=21.95%** π=41.85% ρ=22.57% 3016	**π=34.29%** **ρ=17.96%** π=30.79% ρ=16.08% 3007	π=36.96% ρ=18.13% **π=38.14%** **ρ=17.37%** 3016	π=39.29% ρ=19.89% **π=41.58%** **ρ=20.81%** 3011	**π=42.27%** **ρ=28.96%** π=40.1% ρ=26.22% 8276
Pred LinkTags	**π=49.43%** **ρ=21.95%** π=41.85% ρ=22.57% 3016	π=34.16% ρ=21.86% π=34.16% ρ=21.86% 2940	**π=35.53%** **ρ=19.86%** π=24.26% ρ=16.6% 2941	π=27.21% ρ=17.85% **π=30.5%** **ρ=17.6%** 3001	π=29.66% ρ=19.89% **π=31.18%** **ρ=20.56%** 2954	**π=42.76%** **ρ=29.18%** π=41.63% ρ=27.88% 8276
PredList Headings	**π=34.29%** **ρ=17.96%** π=30.79% ρ=16.08% 3007	**π=35.53%** **ρ=19.86%** π=24.26% ρ=16.6% 2941	π=30.7% ρ=19.42% π=30.7% ρ=19.42% 1644	π=25.11% ρ=17.58% **π=27.63%** **ρ=17.29%** 2832	π=25.7% ρ=17.18% **π=28.76%** **ρ=16.03%** 2402	**π=41.85%** **ρ=28.53%** π=39.42% ρ=26.96% 8276
Pred Headings	π=36.96% ρ=18.13% **π=38.14%** **ρ=17.37%** 3016	π=27.21% ρ=17.85% **π=30.5%** **ρ=17.6%** 3001	π=25.11% ρ=17.58% **π=27.63%** **ρ=17.29%** 2832	π=25.62% ρ=16.64% π=25.62% ρ=16.64% 2828	π=24.5% ρ=17.1% **π=26.62%** **ρ=17.19%** 2911	**π=41.72%** **ρ=28.36%** π=40.76% ρ=27.28% 8276
PredLink Paragraph	π=39.29% ρ=19.89% **π=41.58%** **ρ=20.81%** 3011	π=29.66% ρ=19.89% **π=31.18%** **ρ=20.56%** 2954	π=25.7% ρ=17.18% **π=28.76%** **ρ=16.03%** 2402	π=24.5% ρ=17.1% **π=26.62%** **ρ=17.19%** 2911	π=27.79% ρ=16.86% π=27.79% ρ=16.86% 1143	**π=41.51%** **ρ=28.86%** π=41.02% ρ=28.37% 8276
Own Text	**π=42.27%** **ρ=28.96%** π=40.1% ρ=26.22% 8276	**π=42.76%** **ρ=29.18%** π=41.63% ρ=27.88% 8276	**π=41.85%** **ρ=28.53%** π=39.42% ρ=26.96% 8276	**π=41.72%** **ρ=28.36%** π=40.76% ρ=27.28% 8276	**π=41.51%** **ρ=28.86%** π=41.02% ρ=28.37% 8276	π=42% ρ=29.13% π=42% ρ=29.13% 8276

Table B.31: WebKB Tagging Round Robin Meta Predecessor -WebKB Merging Round Robin Meta Predecessor

Merging outperforms Tagging in a bit more than half of the cases.

	Words Around	Pred LinkTags	PredList Headings	Pred Headings	PredLink Paragraph	Own Text
Words Around	π=39.56% ρ=20.52% π=39.56% ρ=20.52% 3006	**π=51.6%** **ρ=24.26%** π=39.63% ρ=22.72% 3016	**π=32.95%** **ρ=19.99%** π=28.91% ρ=18.2% 3007	**π=27.39%** **ρ=17.03%** π=23.98% ρ=16.18% 3016	**π=40.38%** **ρ=22.64%** π=39.82% ρ=22.76% 3011	**π=37.23%** **ρ=25.37%** π=18.88% ρ=15.4% 8276
Pred LinkTags	**π=51.6%** **ρ=24.26%** π=39.63% ρ=22.72% 3016	π=41.23% ρ=29.74% π=41.23% ρ=29.74% 2940	**π=39.01%** **ρ=26.41%** π=34.83% ρ=25.3% 2941	π=30.63% ρ=19.05% **π=32.75%** **ρ=18.81%** 3001	π=33.85% ρ=22.74% **π=36.25%** **ρ=22.26%** 2954	π=39.78% ρ=24.96% **π=39.87%** **ρ=21.47%** 8276
PredList Headings	**π=32.95%** **ρ=19.99%** π=28.91% ρ=18.2% 3007	**π=39.01%** **ρ=26.41%** π=34.83% ρ=25.3% 2941	π=24.44% ρ=17.22% π=24.44% ρ=17.22% 1644	π=26.04% ρ=17.45% **π=30.31%** **ρ=18.64%** 2832	**π=26.04%** **ρ=16.54%** π=24.18% ρ=15.73% 2402	**π=38.55%** **ρ=26.67%** π=13.71% ρ=14.99% 8276
Pred Headings	**π=27.39%** **ρ=17.03%** π=23.98% ρ=16.18% 3016	π=30.63% ρ=19.05% **π=32.75%** **ρ=18.81%** 3001	π=26.04% ρ=17.45% **π=30.31%** **ρ=18.64%** 2832	π=22.8% ρ=16.05% π=22.8% ρ=16.05% 2828	π=24.46% ρ=16.9% **π=29%** **ρ=17.17%** 2911	**π=35.23%** **ρ=21.44%** π=14.97% ρ=14.5% 8276
PredLink Paragraph	**π=40.38%** **ρ=22.64%** π=39.82% ρ=22.76% 3011	π=33.85% ρ=22.74% **π=36.25%** **ρ=22.26%** 2954	**π=26.04%** **ρ=16.54%** π=24.18% ρ=15.73% 2402	π=24.46% ρ=16.9% **π=29%** **ρ=17.17%** 2911	π=28.86% ρ=19.2% π=28.86% ρ=19.2% 1143	**π=42.39%** **ρ=28.29%** π=26.25% ρ=17.52% 8276
Own Text	**π=37.23%** **ρ=25.37%** π=18.88% ρ=15.4% 8276	π=39.78% ρ=24.96% **π=39.87%** **ρ=21.47%** 8276	**π=38.55%** **ρ=26.67%** π=13.71% ρ=14.99% 8276	**π=35.23%** **ρ=21.44%** π=14.97% ρ=14.5% 8276	**π=42.39%** **ρ=28.29%** π=26.25% ρ=17.52% 8276	π=- ρ=- π=- ρ=- 8276

Table B.32: WebKB Tagging Round Robin Hyperlink Ensembles -WebKB Merging Round Robin Hyperlink Ensembles
The results are equally distributed: Each method between Merging and Tagging wins one-half of the matches.