Ensemble-style Self-training on Citation Classification

Cailing Dong **Ulrich Schäfer**
German Research Center for Artificial Intelligence (DFKI) GmbH
Language Technology Lab, Campus D 3 1, D-66123 Saarbrücken, Germany
`cherinedong@gmail.com` `ulrich.schaefer@dfki.de`

Abstract

Classification of citations into categories such as use, refutation, comparison etc. may have several relevant applications for digital libraries such as paper browsing aids, reading recommendations, qualified citation indexing, or fine-grained impact factor calculation. Most citation classification approaches described so far heavily rely on rule systems and patterns tailored to specific science domains. We focus on a less manual approach by learning domain-insensitive features from textual, physical, and syntactic aspects. Our experiments show the effectiveness of this feature set with various machine learning algorithms on datasets of different sizes. Furthermore, we build an ensemble-style self-training classification model and get better classification performance using only few training data, which largely reduces the manual annotation work in this task.

1 Introduction

Still in the current age of Semantic Web, structured repositories, ontologies and huge databases, scientific knowledge is preserved and transported in rather unstructured pieces of information, vulgo scientific papers or other similar, textual representations. Except bibliographical metadata, (manually) assigned keywords or coarse-grained classifications, there is no further, general method to unlock the written treasures without tedious reading, while full-text search is generally considered too imprecise and may be misleading.

Citations, spread over scientific text, are important anchors that help to structure the broad publication space. They are of invaluable importance to beginners in a scientific field as they ultimately point to seminal, original work and knowledge not explicitly available or repeated in every publication. Citations are also the primary discourse links in scientific discussion which typically span over years or even decades. Furthermore, citations are helpful to understand and reproduce findings. Thus, they form a predominant text feature for every reader. Wan et al. (2009), e.g., present a study on user needs for browsing scientific publications and show that citations play an important role, Schäfer and Kasterka (2010) suggest a novel user interface for navigating in typed citation graphs.

Garfield (1965) is probably the first to discuss an automatic computation of a citation classification. Many studies on citation classification are generally derived from the four-dimensional citation schema proposed by Moravcsik and Murugesan (1975). They distinguish between confirmative vs. negational; conceptual (theory) vs. operational (method); evolutionary (build on cited work) vs. juxtapositional (alternative to cited work); and organic (necessary to understand, reproduce) vs. perfunctory (citation out of politeness, policy, piety). The number of different proposed classes (also called citation *functions*) varies from 3 to 35 (Garfield, 1965; Garzone, 1996; Mercer and DiMarco, 2004; Teufel et al., 2006; Harwood, 2009). Most approaches try to identify more detailed dimensions and mutually exclusive classes by making use of many different features, such as the location of the *citation sentence*[1], surrounding POS tags, and the Boolean information indicating self-citation. In (Teufel et al., 2006), POS tags were employed to find grammatical subjects and further classified as specific agent types, while Mercer and DiMarco (2004) proved the efficiency of rhetoric cues in citation classification. Both studies suggest the potential capability of syntactic features in classifying citations.

Differently from previous work that mainly focuses on cue words and depends on large training sets, our work has the following contributions:

1. *Small but comprehensive feature set:* we only use a small set of features to describe a cita-

[1]We call the exact single sentence which contains the citation the *citation sentence*. There may be several citation sentences for each *reference* listed at the end of a paper.

tion sentence. It covers textual, physical and syntactic aspects and is domain-independent, which could help to make the adaptation to different science domains minimal.

2. *Robust supervised classification model:* we compare the performance of different machine learning algorithms with various sizes of training data. The results show the robustness of the classifiers on our feature set even with a small training set.
3. *Ensemble-style self-training classification model:* we design a semi-supervised learning algorithm to make use of unlabeled data. Differently from standard self-training algorithms, we use an ensemble learning model to choose more reliable new labeled data and achieve good performance. This approach helps to reduce the manual annotation effort.

The paper is structured as follows. We describe our corpus and annotation schema in Section 2, the definition of features will be elaborated in Section 3. We explain the classification experiments in Section 4 and finally conclude in Section 5.

2 Corpus and Annotation

2.1 Definition of Citation Functions

In this paper, we focus on the dimension of "organic or perfunctory" citations in Moravcsik and Murugesan (1975)'s schema and divide the citations into the following four general categories:

- **Background *(class0)***: citations which describe background of the main topic on the whole, or provide recent studies and state-of-the-art approaches in a general way.
- **Fundamental idea *(class1)***: citations about main previous work which inspired or gave specific hints on the current work.
- **Technical basis *(class2)***: citations of important tools, methods, data and other resources used or adapted in the current work.
- **Comparison *(class3)***: citations comparing methods or results with the current work.

We build these citation categories mainly due to the following reasons. First of all, these categories cover the most general and mutually exclusive citation functions. One could easily sketch a picture of a typical, arbitrary scientific publication based on citations from these categories. Secondly, rhetorical and syntactic characteristics are comparatively obvious in terms of functions in this basic level. Thus, our strategy might be valuable for the construction of further detailed automatic citation classification models with more fine-grained categories.

2.2 Corpus

Our corpus comes from the ACL (Association for Computational Linguistics) Anthology[2] (Bird et al., 2008), a comprehensive collection of scientific conference and workshop papers in the area of computational linguistics and language technology. We randomly chose papers from proceedings of the ACL conference in 2007 and 2008. Detailed information on our corpus is provided in Table 1. Corpus pre-processing and the annotation schema will be elaborated in the next subsections.

2.3 Corpus Pre-processing

For each paper in the corpus, we extract citation and reference information from the original PDF file by employing the citation parsing tool ParsCit (Councill et al., 2008). The *citation text* (a text snippet surrounding a citation) delivered by ParsCit is usually not sensitive to paragraph and sentence boundaries and may contain noise such as page number and footnotes. Therefore, we designed a text parser to get better citation sentences and related information by the following steps:

1. From the HTML output of a commercial PDF-to-text extraction tool, we generate a well-organized file with explicit paragraph and sentence boundaries.
2. We extract the titles of each section and its corresponding subsections, and map them into one of six predefined *section categories*[3]: ***Introduction***, ***Related work***, ***Method***, ***Experiment***, ***Evaluation***, and ***Conclusion***. We assign each sentence in the paper its corresponding section category.
3. We extract the corresponding correct citation sentence for a given reference based on its *citation marker* (denoted by authors and publishing year) and *citation text* from ParsCit.

2.4 Annotation Schema

Two annotators annotated the papers independently following our guidelines. The annotators not only focused on each citation sentence separately, but also read the paragraph and the whole section where the sentence is located, then made

[2]`http://aclweb.org/anthology`

[3]We associate section categories by means of synonyms which usually occur in the corresponding section titles.

ACL paper ID	# of distinct citation sentences	citation function distribution (class0 : class1 : class2 : class3)	remarks
P08-1009 ∼ P08-1050	731	485 : 160 : 52 : 34	long-paper set1
P07-1001 ∼ P07-1050	784	492 : 196 : 67 : 29	long-paper set2
P08-2001 ∼ P08-2030	253	173 : 65 : 8 : 7	short-paper set

Table 1: Annotated corpus

a decision on the function of the citation over the whole paper, trying to be in the same perspective as the authors. The inter-annotator agreement between these two annotators is 0.757 measured by Cohen's kappa coefficient (Cohen, 1960), with parameter $n = 4$, $N = 1768$, and $k = 2$. This is quite high given the fact that a kappa value of 0.69 is considered as marginally stable, and 0.8 is considered as stable (Teufel et al., 2006).

3 Feature Set Construction

In this work, we consider the features of each citation sentence in three views: textual, physical and syntactic.

3.1 Textual Features

From the perspective of natural language processing, without any external information, the words in a citation sentence provide the textual distinction in classification. That's the reason why most of the previous work on citation classification is dominated by cue words. We also extract cue words representing certain citation functions, which are listed in Table 2.

From the table, one can observe that firstly only a few cue words exist in each group. Secondly, most cue words in the same group are synonyms. Thirdly, these cue words are domain-independent. We define subject$_{cue}$ to indicate if the agent of the action described in the given citation sentence is related to the current work. It plays an important role in improving the distinguishing capability of other cue words. Besides, as mentioned before, DiMarco et al. (2006) showed the frequency of hedging cues in citation contexts and their importance in citation classification. We also observe that some hedging words listed in hedge$_{cue}$ and suggest$_{cue}$ occur more often in the class Background compared to other classes.

The textual features we defined are as follows:

- a Boolean feature for each group in Table 2, which indicates the existence of *any* cue word from the group in the citation sentence.
- a numerical weight for each group in Table 2, denoting the number of cue words from the group occurring in the citation sentence.
- a Boolean feature and corresponding weight feature for subject$_{cue}$ in *neighbor sentences*[4]. Normally, the more frequently subject$_{cue}$ occurs in consecutive sentences, the more likely the corresponding citation sentence is related to the current work.

3.2 Physical Features

As observed by other researchers on this topic, e.g. Teufel et al. (2006), the sentence location is an important feature, since it might be the most reliable information on citation function one could obtain from the paper directly. Specifically, we take the following physical features into account to distinguish between different citation functions:

- *Location*: section category, belonging to one of the predefined six categories described in Section 2.3.
- *Popularity*: number of references cited in the same sentence.
- *Density*: number of different references in the citation sentence and its neighbor sentences.
- *AvgDens*: average of *Density* among neighbor sentences surrounding the citation sentence.

3.3 Syntactic Features

During annotation, we found that the sentence structure of citation sentences varies according to their function. For example, citation sentences describing background of work are usually in active voice, while basic methods or tools used in the papers are in most cases introduced in passive voice. When the authors review some previous work in a general way, they tend to use present perfect tense, while using simple present tense to elaborate on their own work. These syntactic and writing styles can be extracted based on the POS sequences of citation sentences. In our work, the POS tags are generated by TreeTagger (Schmid, 1994), trained on the Penn Treebank (Marcus et al., 1994).

The typical syntactic patterns we defined are listed as follows. All examples are extracted from papers in our corpus, and the source paper id (ACL

[4]We define *neighbor sentences* as the sentences right before and after the given citation sentence.

Function	Group	Cue words
	subject$_{cue}$	we, our, us, table, figure, paper, algorithm, here
Background *(class0)*	**quantity**$_{cue}$	many, some, most, several, number of, numerous, variety, range of
	frequency$_{cue}$	usually, often, common(ly), typical(ly), traditional(ly)
	tense$_{cue}$	recent(ly), prior, previous, early
	example$_{cue}$	such as, for example, for instance, e.g.
	suggest$_{cue}$	may, might, could, would, will, can, should
	hedge$_{cue}$	suppose, conjecture, want, possible
Fundamental idea *(class1)*	**idea**$_{cue}$	following, similar to, motivate, inspired, idea, spirit
Technical basis *(class2)*	**basis**$_{cue}$	provided by, taken from, extracted from, based on, use, run, apply, extend, measure, evaluate, modify, extract
Comparison *(class3)*	**compare**$_{cue}$	compare, differ, deviate, contrast, exceed, outperform, opposed, consistent with, significant
	result$_{cue}$	result, accuracy, precision, performance, baseline

Table 2: Group of cue words corresponding to citation functions

ID) is denoted in square brackets following each example sentence. We also mark the text corresponding to each specific syntactic structure by underlines. For the POS sequence of each citation sentence, we delete the citation marker in parentheses for convenience.

1. **“.*\\(\\) VV[DPZN].*”:** *Fox <u>() showed</u> that cohesion is held in the vast majority of cases for English-French* [P08-1009]
2. **“.*(VHP|VHZ) VV.*”:** *while Cherry and Lin () <u>have shown</u> it to be a strong feature for word alignment* [P08-1009]
3. **“.*VH(D|G|N|P|Z) (RB)*VBN.*”:** *Inducing features for taggers by clustering <u>has been</u> tried by several researchers ().* [P08-1048]
4. **“.*MD (RB)*VB(RB)* VVN.*”:** *For example, the likelihood of those generative procedures <u>can be accumulated</u> to get the likelihood of the phrase pair ().* [P08-1010]
5. **“[^IW.]*VB(D|P|Z) (RB)*VV[ND].*”:** *Our experimental set-up <u>is modeled</u> after the human evaluation presented in ().* [P08-1009]
6. **“(RB)*PP (RB)*V.*”:** *<u>We use</u> Conditional Random Fields (CRFs) () to perform this tagging.* [P08-1047]
7. **“.*VVG (NP)*(CC)*(NP).*”:** *<u>Following ()</u>, we provide the annotators with only short sentences: those with source sentences between 10 and 25 tokens long.* [P08-1009]

Figure 1 illustrates the distribution of these patterns in different classes (we consider class1 and class2 together, denoted as class1+2) in long-paper set1 and short-paper set (long-paper set2 is used separately for further validating the feature sets). Among these syntactic patterns, the figures showed that the distribution of these patterns in the set of short papers and the set of long papers are consistent, i.e., the first 4 patterns are representative for class Background, while the other 3 patterns are typical for the other 3 classes.

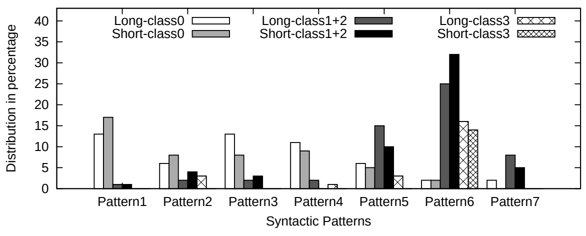


Figure 1: Distribution of syntactic patterns

We define a Boolean feature for each syntactic pattern indicating whether the pattern matches the POS sequence of the citation sentence. Combined with the cue words defined in Table 2, especially with subject$_{cue}$ and typical verbs in class1 to class3, the contribution of these patterns to overall performance increases significantly.

Thus, our final feature set is the collection of these textual, physical and syntactic features.

4 Citation Classification

4.1 Effectiveness of Syntactic Features

First of all, we build a set of experiments to test the effectiveness of syntactic features in citation classification. We compare the classification performance of our feature set with and without syntactic features on different machine learning algorithms, and with various training-testing ratios. We employ the meta class *AttributeSelectedClassifier* in Weka (Hall et al., 2009), and set *ChiSquaredAttributeEval* and *Ranker* (threshold 0) as evaluator and search method for attribute selection. The following 5 machine learning algorithms provided in Weka are employed as basic classifiers: *BayesNet*, *NaiveBayes*, *SMO* (*PolyKernel* is chosen as support vector kernel), *J48* and *RandomForest*.

We randomly split our whole corpus (including long-paper set2) into training dataset and test dataset with training instance ratios of 80%, 60%, 20% and 10%, respectively. For each of these four training-testing ratio configurations, we use 20 different random seeds to generate 20 different datasets. With each dataset, we measure the testing result by Macro-F (the mean of the F-measures of all classes), following Teufel et al. (2006). Then the final performance is measured by the average Macro-F of these 20 experiments with the same training-testing ratio configuration.

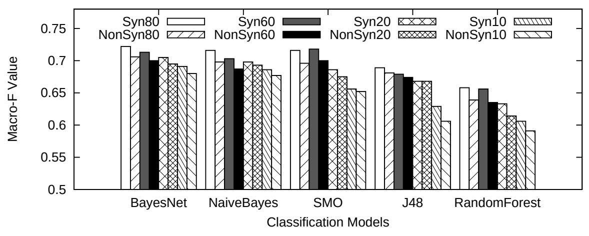


Figure 2: Performance of feature sets with and without syntactic features

Figure 2 illustrates the performance of feature sets with and without syntactic features on different sizes of training data. *Syn80* and *NonSyn80* denote feature sets with and without syntactic features when the ratio of training data is 80% respectively, analogously for other training data ratios. We can see that the feature sets with syntactic features can always beat the ones without syntactic features for all the datasets with different sizes of training data, independently of the classification model used. Besides, for all these datasets in different configurations, BayesNet and NaiveBayes are not very sensitive to the size of the training data. That is, these two supervised classification models can well learn our feature sets and make a good classification performance with only few training data (training ratio as 10% here, about 170 instances). It proves the effectiveness of our feature set to some extent.

4.2 Ensemble-style Self-training for Citation Classification

As in many other natural language processing tasks, large training datasets need to be annotated manually in order to improve the performance of citation classification. That is, a good annotation schema and abundant annotated training data are the basis for building a good automatic citation classification model. Because annotation is time-consuming, we design a semi-supervised learning algorithm to make use of unlabeled data under small training datasets in citation classification. Specifically, we adopt the idea of self-training (Zhu and Goldberg, 2009) to employ unlabeled data, but select new labeled data in the process of self-training in a more reliable way by taking advantage of the ensemble learning (Opitz and Maclin, 1999) algorithm.

4.2.1 Basic Self-training Algorithm

As a typical algorithm in semi-supervised learning, the self-training algorithm tries to extend the training dataset by changing some unlabeled data to labeled data with its own current predicted labels. We describe the basic self-training process in detail in Algorithm 1[5]. There are two main approaches to select the unlabeled data: one is to select the instances with highest predictive confidence *(prob-selfTrain)*, which is the original approach; the other is to randomly select some instances from unlabeled dataset *(random-selfTrain)*, which has been shown to be effective in some applications. As shown in line 6 and line 7, we implement these two approaches in parallel. Three different basic classifiers are used in this algorithm. From the main procedure of this algorithm, one can see that each classifier maintains its own labeled and unlabeled datasets in each iteration independently. This algorithm is set to be our baseline algorithm.

4.2.2 Ensemble-style Self-training Algorithm

Every supervised learning algorithm tries to search for the best hypothesis through a hypothesis space, therefore makes good predictions with a particular problem, i.e, based on given training data. Taking advantage of this property, ensemble learning tries to improve the prediction accuracy by combining different supervised learning algorithms. In self-training, each prediction process is actually supervised learning, therefore the quality of the training data is very important. Furthermore, self-training is based on the assumption that one's own high confidence predictions are correct, i.e. the final results could be worse if the model repeatedly learns from its own previous wrong decisions. To avoid such a situation, we devise an ensemble-style self-training algorithm by using ensemble learning to choose reliable new labeled

[5]Set $\{a,b,c,d\}$ denotes the number of instances in class0, class1, class2 and class3 as a, b, c, d, respectively. Analogously for set $\{m,n,p,q\}$ for indicating the number of added instances in each iteration.

Algorithm 1 Baseline: basic self-training algorithm

```
Require:
        Whole dataset $D$ ;   training set ratio $\alpha$ ;   Basic classifiers $Base1$, $Base2$, $Base3$;
        Iteration times $K$;   Initial number of labeled instances $\{a,b,c,d\}$;
        Number of instances added to labeled dataset in each loop: $\{m,n,p,q\}$;
Pre-process:
        Training dataset $Train$ with number of instances $|D| * \alpha$;   Test dataset $Test$ with instances $\{D\} - \{Train\}$;
        Labeled dataset $L$ with initial number of instances $\{a,b,c,d\}$ randomly selected from $Train$;
        Unlabeled dataset $U$ with instances $\{Train\} - \{L\}$;
        Labeled dataset $L_1 = L$, $L_2$=$L$ and $L_3$=$L$;   Unlabeled dataset $U_1 = U$, $U_2$=$U$ and $U_3$=$U$;
Procedure:
 1: for $k = 0$ to $K$ do
 2:     for each classifier $Base_i$ $(i = 1,2,3)$ do
 3:         Use $L_i$ to train classifiers $Base_i$
 4:         Obtain the test results on $Test$ by classifier $Base_i$
 5:         Make prediction for each instance in $U_i$ by classifier $Base_i$
 6:         a. prob-selfTrain: Select $\{m,n,p,q\}$ instances which have highest prediction confidence from $U$, denoted as $L_{i-add}$
 7:         b. random-selfTrain: Randomly select $\{m,n,p,q\}$ instances from $U$, denoted as $L_{i-add}$
 8:         $L_i = L_i + L_{i-add}$;   $U_i = U_i - L_{i-add}$
 9:     end for
10: end for
```

Algorithm 2 Ensemble-style self-training algorithm

```
Require: configuration of $Train$, $Test$, $L$, $U$, and value of parameters $\{a,b,c,d\}$, $\{m,n,p,q\}$, $K$ as well as basic classifiers
    $Base1$, $Base2$, $Base3$ are the same as those in Algorithm 1;
Procedure:
 1: for $k = 0$ to $K$ do
 2:     Use $L$ to train classifiers $Base1$, $Base2$, $Base3$, respectively
 3:     Obtain the test results on $Test$ by classifier $Base1$, $Base2$, $Base3$ respectively
 4:     Obtain the test results on $Test$ by ensemble model $Ensemble$, consist of $Base1$, $Base2$ and $Base3$
 5:     Make prediction for each instance in $U$ by $Ensemble$
 6:     Randomly select $\{m,n,p,q\}$ instances from $U$, denoted as $L_{add}$
 7:     $L = L + L_{add}$;   $U = U - L_{add}$
 8: end for
```

data in each iteration. The detailed procedure is displayed in Algorithm 2. Here, the datasets and parameters are set to be the same as those in Algorithm 1, in order to compare the performance of both algorithms. In this ensemble-style self-training algorithm, one can see that all the basic classifiers and ensemble model *Ensemble* share the same labeled dataset L and unlabeled dataset U, which is maintained by *Ensemble* via choosing new labeled data in ensemble way (line 5 and 6).

4.2.3 Ensemble Learning

Normally, there are two main combination rules in ensemble learning to make the final prediction for a given instance. One is probability-based, that is, choosing the decision of the basic classifier which produces the highest prediction confidence for the instance. Another is to use majority voting. In our experiments, we choose BayesNet, SMO and NaiveBayes as the basic classifiers in our ensemble model and adopt the majority voting combination rule, due to the following reasons: 1) these three models have better overall performance according to Figure 2; 2) Usually, the more diverse the member algorithms, the more effective their ensemble model can be (Rokach, 2010). The classification mechanisms of BayesNet and SMO are totally different, which can ensure the performance of the ensemble model to some extent; 3) it is impossible to use probability-based combination rules due to the uninterpretable prediction probability in SMO. Thus, using a classifier with good overall performance such as NaiveBayes in majority vote could ensure the accuracy of ensemble learning to a certain degree.

4.2.4 The Class Imbalance Problem

From previous research on citation classification, independently of how many different functions were defined, one can observe that the dataset is class-imbalanced. For example, in a sentiment-oriented classification schema, negative citations are much less frequent than positive citations. Conversely, a high volume of citations actually is used to describe the background. Thus, class imbalance is actually a key issue in automatic citation classification. Previous research often ignored this fact, possibly because pattern-driven or rule-based models are less sensitive to such a distribution of datasets. Usually, standard classifiers try to get the overall accuracy with the cost of misclassifying the instances in small classes (Zhou and

Liu, 2006), while small classes are always those classes people are most interested in. Taking our citation classification as an example, "organic" citations are more attractive than the "perfunctory" ones. In our experiments, we try to balance the instances of these classes to protect small classes.

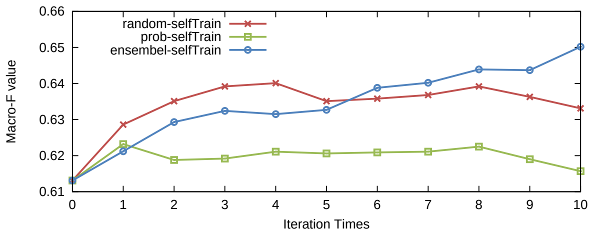
4.2.5 Experiments

In this set of experiments, we also wrap these three base classifiers with the meta class *AttributeSelectedClassifier* in the same configuration as in the last experiments in Section 3.1. As listed in Table 1, the average ratios among these 4 classes are imbalanced, around *16:6:1.8:1*. On the one hand, we try to keep the natural distribution to some extent (number of instances in the two bigger classes: class0:class1≈2:1; number of instances in the two smaller classes: class2:class3≈2:1). On the other hand, we slightly reduce the intensity of the class imbalance to avoid misclassifying too many instances from the small classes (here, we limit the ratio of class0 and class3 to be less than 5). To protect the small classes, we set the distribution of the initial labeled dataset to around *5:2.5:2:1*. A few new labeled data with similar distributions are added in each iteration to avoid introducing too much noise. We conduct two experiments.

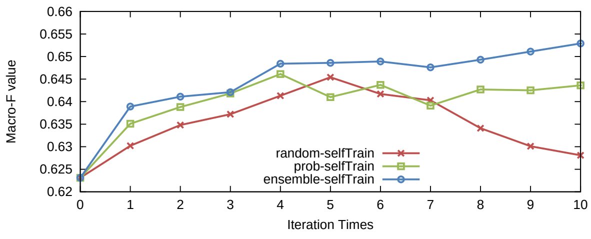
Exp-a: We set the number of instances in the smallest class class3 as 5, in order to ensure the accuracy. The iteration time K is 10. So α is set to be 0.3 in case insufficient unlabeled data to be added as labeled data. That is, 30% of the instances are randomly extracted as training set and the remaining instances are set as test data. Thus, according to our predefined class distribution, we set initial labeled data $\{20, 10, 8, 5\}$, and the rest of the training data are regarded as unlabeled data. The number of new labeled data in each iteration is $\{6, 4, 3, 1\}$.

Exp-b: In this experiment, we use 3 labeled instances in class3 to conduct the self-training process, initial labeled data $\{a, b, c, d\}$ is set to be $\{15, 8, 6, 3\}$ following predefined class distribution. Less new labeled data is added each time, $\{m, n, p, q\}$=$\{5, 3, 2, 1\}$ with more iterations, K=30. α is set to 0.3 due to the long iterations.

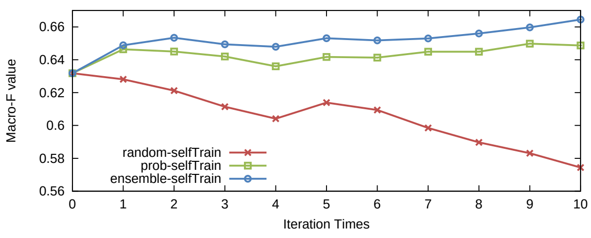
As mentioned before, self-training has a high demand on training data, especially initial labeled data. That is, the quality of the initial labeled data can largely affect the final overall performance. In order to obtain fairly general results, we repeat each experiment described above 10 times



(a) SMO



(b) BayesNet

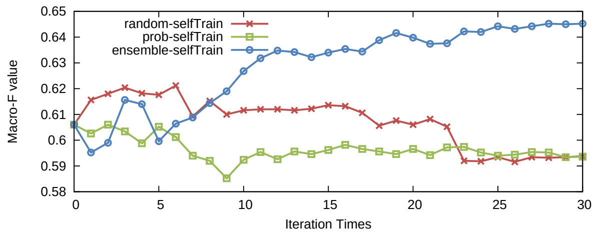


(c) NaiveBayes

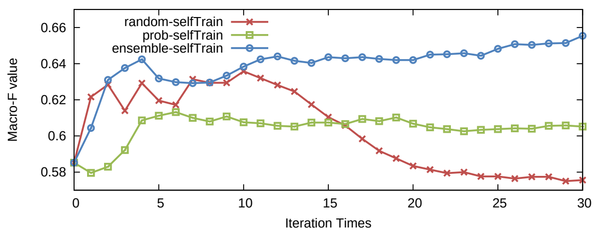
Figure 3: *Exp-a:* Basic self-training vs. ensemble-style self-training with different basic classifiers

with different seeds, splitting the dataset into training set and test set and therefore the initial labeled data each time are different. The final results are measured by the average value of Macro-F in these ten experiments. Figure 3 and Figure 4 illustrate the final average Macro-F values obtained by different basic classifiers with parameters given in *Exp-a* and *Exp-b*, respectively. Here, Macro-F in iteration 0 is obtained based on initial labeled data, so the algorithms with *prob-selfTrain*, *random-selfTrain* and ensemble-style self-training (denoted as *ensemble-selfTrain*) start at the same point for the same basic classifier.

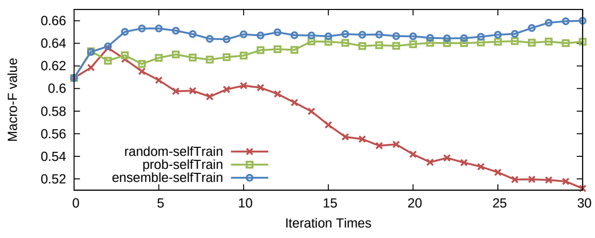
From Figure 3 and Figure 4, we can observe that for the two probability-driven classifiers BayesNet and NaiveBayes, their *prob-selfTrain* process is better than *random-selfTrain* on the whole. We observe the opposite for classifier SMO since its prediction confidence is meaningless. In general, our ensemble-style self-training algorithm outperforms both kinds of basic self-training algorithms independently of the basic classifier used. Its self-training process is quite stable due to the relatively reliable new labeled data ensured by ensemble learning strategies. Among all three basic clas-



(a) SMO



(b) BayesNet



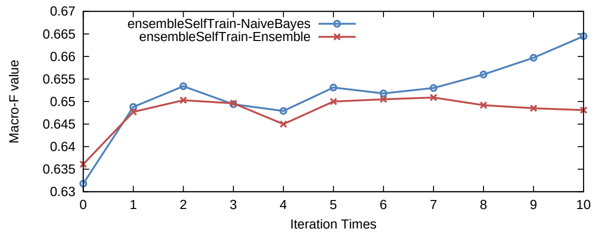
(c) NaiveBayes

Figure 4: *Exp-b:* Basic self-training vs. ensemble-style self-training with different basic classifiers

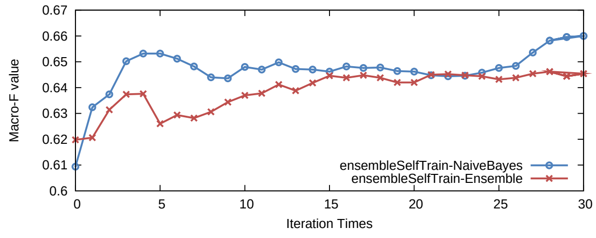
sifiers with the same experimental setup, NaiveBayes has the best performance not only at the starting point (iteration 0), but also in the whole ensemble-style self-training process.

In our ensemble-style self-training algorithm, we also build an ensemble model *Ensemble*. In Figure 5, we compare the performance of ensemble-style self-training with basic classifier *Ensemble* (denoted as *ensembleSelfTrain-Ensemble*) and NaiveBayes under experimental setup *Exp-a* and *Exp-b*, respectively.

Based on the observation, we found that although *Ensemble* is a good guide on selecting new labeled data, its self-training performance is not as good as NaiveBayes. There might be two main reasons. First of all, without self-training, NaiveBayes itself fits our feature sets quite well and shows the best performance among all the classifiers. Combining with other classifier as done in model *Ensemble* could affect its good performance. Secondly, ensemble models tend to be over-fitting to the training data. During the self-training process, the training data changes slightly by adding few new labeled data each time, so the over-fitting problem occurs soon. Thus, we choose



(a) *Exp-a*



(b) *Exp-b*

Figure 5: Ensemble-style self-training on NaiveBayes vs. *Ensemble*

ensemble-style self-training with main basic classifier NaiveBayes as our final classifier for citation classification.

5 Summary

In this work, we defined a citation classification schema to better sketch the skeleton of a scientific paper from its background, fundamental ideas, technical basis and performance comparison. These citation functions are distinguished by features from textual, physical and syntactic aspects, which are simple but comprehensive, and domain-independent. Using different supervised learning classifiers on various sizes of training datasets, we improved and demonstrated the efficiency of our feature set by adding syntactic patterns extracted from POS tags. The supervised classification models NaiveBayes and BayesNet have shown their robustness on our feature set, with only about 170 training instances. Furthermore, we built an ensemble-style self-training algorithm to better use and extend training data. Using about only 40 labeled instances, our final classifier can improve the Macro-F value by almost 5%. This model could largely alleviate manual annotation in citation classification.

Acknowledgments

This research has been funded by the German Federal Ministry of Education and Research under contract 01IW08003 (project TAKE). We thank the anonymous reviewers for insightful comments.

References

Steven Bird, Robert Dale, Bonnie Dorr, Bryan Gibson, Mark Joseph, Min-Yen Kan, Dongwon Lee, Brett Powley, Dragomir Radev, and Yee Fan Tan. 2008. The ACL anthology reference corpus: A reference dataset for bibliographic research. In *Proceedings of the Language Resources and Evaluation Conference (LREC-2008)*, pages 1755–1759, Marrakesh, Morocco.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46.

Isaac G. Councill, C. Lee Giles, and Min-Yen Kan. 2008. ParsCit: An open-source CRF reference string parsing package. In *Proceedings of the Language Resources and Evaluation Conference (LREC-2008)*, pages 661–667, Marrakesh, Morocco.

Chrysanne DiMarco, Frederick Kroon, and Robert Mercer. 2006. Using hedges to classify citations in scientific articles. In *Computing Attitude and Affect in Text Theory and Applications*, pages 247–263.

Eugene Garfield. 1965. Can citation indexing be automated? In Mary Elizabeth Stevens, Vincent E. Giuliano, and Laurence B. Heilprin, editors, *Statistical Association Methods for Mechanical Documentation*. National Bureau of Standards, Washington, DC. NBS Misc. Pub. 269.

Mark Garzone. 1996. Automated classification of citations using linguistic semantic grammars. Master's thesis, Dept. of Computer Science, The University of Western Ontario, Canada.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1).

Nigel Harwood. 2009. An interview-based study of the functions of citations in academic writing across two disciplines. *Journal of Pragmatics*, 41(3):497–518.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1994. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, 19(2):313–330.

Robert E. Mercer and Chrysanne DiMarco. 2004. A design methodology for a biomedical literature indexing tool using the rhetoric of science. In Lynette Hirschman and James Pustejovsky, editors, *HLT-NAACL 2004 Workshop: BioLINK 2004, Linking Biological Literature, Ontologies and Databases*, pages 77–84, Boston, Massachusetts, USA.

Michael J. Moravcsik and Poovanalingam Murugesan. 1975. Some results on the function and quality of citations. *Social Studies of Science*, 5:86–92.

David Opitz and Richard Maclin. 1999. Popular ensemble methods: an empirical study. *Journal of Artificial Intelligence Research*, 11:169–198.

Lior Rokach. 2010. Ensemble-based classifiers. *Artificial Intelligence Review*, 33:1–39, February.

Ulrich Schäfer and Uwe Kasterka. 2010. Scientific authoring support: A tool to navigate in typed citation graphs. In *Proceedings of the NAACL-HLT 2010 Workshop on Computational Linguistics and Writing*, pages 7–14, Los Angeles, CA.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing*, Manchester, UK.

Simone Teufel, Advaith Siddharthan, and Dan Tidhar. 2006. Automatic classification of citation function. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 103–110, Sydney, Australia.

Stephen Wan, Cécile Paris, Michael Muthukrishna, and Robert Dale. 2009. Designing a citation-sensitive research tool: an initial study of browsing-specific information needs. In *Proceedings of the 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries*, NLPIR4DL '09, pages 45–53, Suntec, Singapore.

Zhihua Zhou and Xuying Liu. 2006. On multi-class cost-sensitive learning. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, pages 567–572.

Xiaojin Zhu and Andrew B. Goldberg. 2009. *Introduction to Semi-Supervised Learning.* Morgan and Claypool.