Combining Active and Ensemble Learning for Efficient Classification of Web Documents

Steffen Schnitzer, Sebastian Schmidt, Christoph Rensing, and Bettina Harriehausen-Mühlbauer

***Abstract*—Classification of text remains a challenge. Most machine learning based approaches require many manually annotated training instances for a reasonable accuracy. In this article we present an approach that minimizes the human annotation effort by interactively incorporating human annotators into the training process via active learning of an ensemble learner. By passing only ambiguous instances to the human annotators the effort is reduced while maintaining a very good accuracy. Since the feedback is only used to train an additional classifier and not for re-training the whole ensemble, the computational complexity is kept relatively low.**

***Index Terms*—Text classification, active learning, user feedback, ensemble learning.**

I. Introduction

DURING the last decade, the Internet has become a main source of information. In November 2013 there were more than 190 million active Web sites online [1]. Since Web sites do not follow any common indexing schema, search engines are the only way to fulfill users' information needs by giving an entry-point to the Web sites with the aimed content. Besides search engines for general purposes like Google[1] or Bing[2], a number of domain specific search engines has evolved over the last years. Those search engines are tailored for the exploration of Web documents from a specific domain. Prominent domains for domain-specific search engines are hotels, restaurants, products or job offers. In contrast to general search engines, these specialized engines provide additional value based on pre-defined knowledge of their respective domains. This knowledge can be used e.g. for offering a faceted search interface, for organising the indexed Web documents or for giving recommendations based on previously viewed Web documents. Since Web documents are normally not annotated with meta information on their content, there is a need to infer this information automatically. One common method for this is the use of machine learning techniques, in particular text classification to identify appropriate class labels from the pre-defined knowledge that match the content. For the use within e.g. a hotel search engine these labels could be the focus of the hotel (*business*, *sports*, *family*, etc.) or for job search engines those could be the field of work (*IT*, *sales*, *medical*, etc.).

Traditional classification approaches require a huge number of manually labeled training instances. In static environments where domains do not adapt over time this results in a large initial effort for human annotators. In dynamic environments where the terminology changes over time, a constant annotation of large number of training instances is required which is not feasible. Hence, there is the need for an efficient solution which provides an excellent classification accuracy with less manual effort compared to traditional machine learning systems and which learns during run-time.

In this paper we present a solution which identifies Web documents that are most helpful for the system's accuracy to be annotated manually and hence to be used for the iterative improvement of the overall text classification system. The solution combines different well-known machine learning techniques such as Ensemble Learning and Active Learning but aims at having fewer time requirements compared to existing solutions.

The remainder of this paper is structured as follows. Section 2 gives an overview on fundamentals and related work in the fields relevant to our work. Based on this, our concept is presented in Section 3. Section 4 presents the methodology and the results of an extensive evaluation with 10,300 Web documents. Our achievements and future work are summarized in Section 5.

II. Fundamentals and Related Work

In this section, we give an overview on important concepts for our work. After a general introduction into the topic of *text classification* and its state-of-the-art we give insights into two general machine learning foundations for our work: *Ensemble Learning* is a technique where various machine learning results are combined into one common result. *Active Learning* allows to incorporate human feedback into a machine learning decision.

A. Text Classification

Text classification describes the automated process of assigning a text one or multiple class labels based on

Manuscript received on December 17, 2013; accepted for publication on February 6, 2014.

Steffen Schnitzer, Sebastian Schmidt, and Christoph Rensing are with Multimedia Communications Lab, Technische Universität Darmstadt, Germany (e-mail: {Steffen.Schnitzer, Sebastian.Schmidt, Christoph.Rensing}@kom.tu-darmstadt.de).

Bettina Harriehausen-Mühlbauer is with the University of Applied Sciences, Darmstadt, Germany (e-mail: Bettina.Harriehausen@h-da.de).

The first two authors contributed equally to this work.

[1]http://www.google.com

[2]http://www.bing.com

characteristics of the text. The class label(s) can describe various attributes of the text such as the topic or the text type. When multiple class labels can be assigned to a single text at the same time, it is referred to as multi-label classification. In our work we face a multi-label classification, but since the class labels are conditionally independent, according to [2] we break it down to a binary classification where we have to decide for each class label if it has to be assigned to a certain text or not.

In the past, a lot of work has been done in the field of text classification with various applications. As for all classification tasks, a model has to be defined first which describes instances to be classified in an abstract way. For the classification of text a widely-used model is the bag-of-words model in combination with the *term frequency-inverse document frequency (TF-IDF)* measure. The bag-of-words model represents text as an un-ordered collection of the occurring words. Since not every word has the same significance for a document, the single words are often weighted. The probably most common weighting scheme is TF-IDF, which assigns weights according to the frequency of a word within the respective text in comparison to all other texts in a corpus [3].

Using each word from a corpus as a feature where the TF-IDF values for single text instances are the feature values results in a high-dimensional space with very sparse vectors. This makes Support Vector Machines (SVMs) the most suitable classification algorithm for text classification [4].

Besides the main goal of an accurate classification, also the timing requirements for training and classification phase have been focus of research. The usage of different parallel classifiers, each of which has been trained on a sub-space of the total classes, has been presented in [5]. This approach outperforms approaches using a single classifier for the whole space of classes in terms of accuracy and speed. It is well suited for text classification tasks with hierarchical class labels but cannot be applied in settings with a large number of classes without a hierarchy.

Different methods have been presented for reducing the human effort for annotation, e.g. Fukumoto et al. present an approach that requires to have only positive examples labeled by humans [6]. More approaches are presented in Section II-C.

B. Ensemble Learning

Ensemble learning allows to combine different machine learning models into a single model. It has been shown that this improves the overall classification accuracy [7]. In this section, we will focus on the technique of *Bagging* since this proved to be most suitable for our problem. We did not employ the technique of *Stacking*, because a single most suitable classification method (SVM) has been identified. The iterative technique of *Boosting* was not used due to time performance reasons, however, the active learning part of our approach bears some characteristics of Boosting. Bagging denotes the idea to apply N instances of the same classification algorithm on N different representative random subsets of the original training set. This results in N different classifier models with different classification results [8]. The resulting labels of the single classifier can then be combined into one common result e.g. via voting or averaging, where voting is more natural for binary classifiers and averaging more natural for classifiers with numeric output. One drawback of this approach is the splitting of the complete training set, which results in smaller training sets for the single classifiers and might hence have a negative impact on the classification accuracy.

C. Active Learning

"The goal of active learning is to minimize the cost of training an accurate model by allowing the learner to choose which instances are labeled for training" [9]. The idea is to let human annotators interactively label the instances with the highest information gain and to improve the classifier by incorporating those instances into a re-training phase.

By doing so, the overall annotation effort is reduced since initially only a small number of instances needs to be annotated and afterwards only the "helpful" instances are added. This concept requires to identify the instances which can improve the classifier substantially. Strategies for selecting the most helpful instances have been focus of research for a while now [10]. Besides the advantages of this concept, also the computational effort needs to be considered. Sophisticated models like the "estimated loss reduction" [11] or the "expected error reduction" [12] are computationally very cost-intensive.

Zhu el al. [13] selects for human feedback the unlabeled instances that change their predicted class label during two consecutive learning steps or which are predicted to be in a certain class with less certainty compared to the previous step.

When making use of the previously introduced technique of Bagging, the result of voting during the classification process can be seen as a measure of certainty about the classification decision. If the single classifiers show to have differing results, the classified instance together with its label can be assumed to be helpful to improve the accuracy of the overall classifier. Li and Snoek present an approach where an ensemble of SVMs for image tag classification is re-trained iteratively with previously miss-classified examples where the correct labels are obtained via crowdsourcing [14]. The authors show that this approach leads to better classification in comparison to re-training with randomly chosen instances. The approach requires that the whole ensemble needs to be re-trained when incorporating new examples.

To conclude, we have seen that various approaches allow for a text classification which is more robust, more efficient or require less human effort. But no combination of these goals has yet been achieved.

III. Concept

A. Overview

In order to be able to use the advantages of ensemble and active learning, a combination of these two methods is sought. To achieve such a combination, two different classifiers are created which depend on each other. On the one hand, there is the ensemble learner, which employs several different classifiers using a voting scheme to find a classification decision. This base classifier is very accurate and represents the effective classification. On the other hand, there is the active learner, which is only trained with documents where the base classifier is very uncertain in its classification decision (ambiguous documents). This active learning classifier is specialized in these ambiguous documents and can be re-trained very fast. We call this combination the *Combined Ensemble and Fast Active Learner (CENFA)* [15].

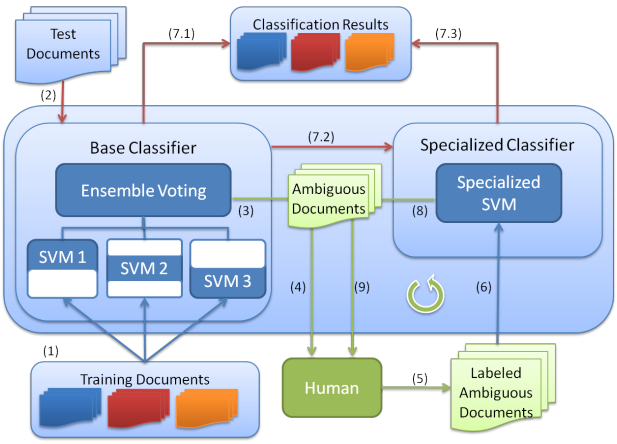


Fig. 1. The *CENFA* classifier

Figure 1 depicts the described concept. The base classifier on the left uses SVMs in a bagged ensemble. The specialized classifier on the right uses a single SVM. Ambiguous documents as identified by the base classifier are labeled by a human and used to train the specialized classifier. Test documents are then classified depending on their ambiguity either according to the results of the base classifier or the results of the specialized classifier. The re-training of the specialized classifier based on the ambiguous documents is performed iteratively.

B. Phases of Training and Classification

The training and classification using *CENFA* is described in detail along Fig. 1 in the following. The process starts with a 2-phased setup and afterwards the regular mode can be employed.

1) Setup Phase 1: At first only the base classifier is trained (1). For that, several *SVMs* are trained on different subsets of the training data acquired by bootstrapping. This application of Bagging produces a very robust initial model.

2) Setup Phase 2: In the following phase, the *CENFA* classifier is provided with classification tasks (2) and the documents to be classified are provided to the different *SVMs*. The base classifier aggregates the different results of the *SVMs* for each document and calculates a decision based on a voting scheme. Based on a confidence threshold, the base classifier decides whether the classified document appears to be ambiguous (3). All non-ambiguous classification results are discarded during this phase. By performing several classification tasks, the number of identified ambiguous documents grows. Those ambiguous documents are then annotated according to human feedback (4) which creates a certain number of labeled ambiguous documents (5). The labeled ambiguous documents are used to train a new classifier which uses a single *SVM* (6). This classifier is trained exclusively with documents that appear ambiguous to the base classifier and is therefore specialized on such documents where the base classifier shows weakness.

3) Regular Mode: Now that the specialized classifier is initially trained, the *CENFA* can enter the regular mode and be used for classification. Documents are first classified by the base classifier (2). For documents that do not appear to be ambiguous, the classification result is output directly (7.1.). When a document appears to be ambiguous to the base classifier, the classification decision is translocated to the specialized classifier (7.2). Now the specialized classifier calculates the decision based on the single *SVM* and populates it (7.3). Here the specialized classifier may identify the document as ambiguous and add it to the ambiguous documents (8). More classification tasks will make the number of ambiguous documents, which are then given to a human for feedback (9), grow again. This is used to re-train the specialized classifier and re-train it iteratively afterwards (5,6,8,9). This leads to steady improvement of the overall classifier during its usage.

This method combines the efficacy of ensemble learning, represented by the bagged base classifier and the efficacy of active learning represented by the fast iteratively trained specialized classifier. Through the simple interfaces, the inner combination of the two classifiers can be hidden and the classifier can be used as a simple active learning classifier.

IV. Evaluation

In order to prove the success of our approach we ran an extensive evaluation from which we present selected results in this section. Before presenting the results themselves, we give insight into the methodology we used for evaluation.

A. Methodology

For evaluation a corpus of 10,300 German Web documents containing job offers was used. The documents do not contain any HTML markup but the pure textual content of the Web sites. Each of these documents was annotated with one or multiple class labels which represent the job offer's respective

TABLE I
CLASSES CONSIDERED FOR EVALUATION TOGETHER WITH THE NUMBER OF POSITIVE AND NEGATIVE EXAMPLES

ID	Name	Positives	Negatives
SD	Software Development	2,077	8,223
TM	Technical Management	1,727	8,573
Sales	Sales	1,587	8,713
P-QA	Production & Quality Assurance	1,501	8,799
TDD	Technical Development & Design	1,069	9,231

field(s) of work. A set of 103 different labels was used for annotation. On average, each instance was annotated with 4.25 labels with a standard deviation of 1.86.

For the purpose of evaluation we considered only the five classes that showed to have the largest number of positive examples, which are instances annotated with the respective class label. This allows to have a large evaluation corpus available. As mentioned above, the multi-label classification problem is solved by building binary classifiers for each single label. The number of positive and negative examples for each classifier are shown in Table I. All instances not annotated with the respective class label are considered as negative training example for this class. Because the evaluation corpus is highly unbalanced, we applied a resampling of the data to achieve a better balanced data distribution. By doing so, we aim to obtain a more robust classifier.

The whole corpus used for the evaluation was preprocessed consistently so that the different classifiers where able to perform their work on the same feature set. To obtain the numeric vectors required for SVM classification, the TF-IDF statistics are gathered for the 10,000 most used words by applying a german tokenizer without using a stop word list or a stemming algorithm.

For simulating the interactive feedback given by the human annotators, we also used parts of this evaluation corpus. For each instance where the classifier decides to request the human for feedback we provide the label from the evaluation corpus. Based on this, we achieve a division of the training data into three sub-sets:

1) Subset *A* is used to train the base classifier.
2) The elements of subset *B* are classified by the base classifier and if an element is identified as ambiguous it is passed to the specialized classifier as training data together with its annotation. All elements that were identified as ambiguous form subset *S*.
3) Subset *C* is used for the evaluation (testing) of the overall CENFA classifier.

Since the goal of our work is a classification approach that can classify instances with the same accuracy as traditional ensemble learning approaches but with reduced manual human effort and with a better timing behavior, we need to compare our approach to other approaches. These approaches will be explained in the following. Subset *S* is the set of ambiguous instances which is used to train the specialized classifier of the CENFA classifier. The *Random* classifier approach uses set *R*, a random selection of elements from subset *B*, to train the specialized classifier. The number of elements in this selection is similar to the number of ambiguous instances the *CENFA* approach uses to train the specialized classifier. In other words, subset *R* is chosen to be of the same cardinality as subset *S*, while both are subsets of set *B*. The aim of this approach is to verify the suitability of using ambiguous instances for incorporating user feedback instead of training the specialized classifier with random instances. In order to examine the benefit of not retraining the base classifier with the ambiguous instances but only the specialized classifier we introduce the *Extended* classifier approach. After having recognized a number of instances as ambiguous, the whole ensemble is re-trained and the accuracy and run time of this approach are compared to the training of *CENFA*'s specialized classifier only. Last but not least, our approach is evaluated against the *Random Single SVM (RSSVM)* approach that uses a single SVM trained with the subset *A* and a random selection from set *B*. It has to be noted that *CENFA*, *Random*, *Extended* and *RSSVM* are all trained with the same amount of training data but the instances used and the overall system architecture vary across these approaches.

The three different classifiers for comparison each have a separate purpose. The *Random* delivers insights on the *accuracy* performance of *CENFA* compared to a classifier which does not use the *active learning* methodology for selecting ambiguous instances. The *RSSVM* delivers insights on *CENFA's accuracy* performance compared to a classifier which does not use the *ensemble learning* methodology and additionally the training time difference to a single *SVM* setup. The *Extended* delivers insights on the *accuracy* performance compared to a classifier which does not apply the provided compromise. Here *CENFA* was expected to be outperformed while being much faster. Table II provides an overview of the different classifiers with their used training sets and their evaluation purpose.

The *CENFA* architecture and the evaluation concept allow to tune different parameters and examine their influence on the overall accuracy in order to determine the best setting. The *dividing factor* denotes the division into the subsets *A*, *B* and *C*; in particular the given number represents the fraction of data that is assigned to subset *A*. Subsets *B* and *C* always hold the same number of instances. Hence, e.g. a *dividing factor* of 0.7 means that *A* consists of 70% of the instances from the evaluation corpus, *B* of 15% and *C* of 15%. A higher *dividing factor* results in a larger training set *A* but a smaller number of instances for the training of the specialized classifier. The second parameter which can be varied is the *confidence value*, which denotes the decision threshold of the ensemble learner up from which an instance is considered as ambiguous. If this is chosen to be very low then only a very small amount of instances from *B* are considered as ambiguous and used for the training of the specialized classifier. Further, the specialized classifier gets only a small amount of instances from *C* assigned for training since the base classifier decides

TABLE II
CLASSIFIERS WITH TRAINING SETS AND EVALUATION PURPOSE

	training sets		
Classifier	base	special	evaluation baseline with the following purpose
CENFA	A	S	proposed approach
Random	A	R	accuracy when using random instead of ambiguous instances
RSSVM	–	A+R	accuracy/timing behavior without ensemble
Extended	A+S	–	accuracy/timing behaviour with complete retraining

on the class for most of the instances. The last parameter which can be tuned is the *number of bagged SVMs* for finding a good trade-off between robustness of the ensemble and accuracy of the single classifiers.

We evaluate the approaches by calculating the *accuracy* of the classification based on a 10-fold cross-validation. Further, we examine the time required for building the classifiers using the different settings. The underlying SVM algorithm's implementation, used by all classifiers during evaluation, applies the *Sequential Minimal Optimization (SMO)* [16] algorithm with the default parameters provided by the Weka[3] framework.

B. Results

The different parameters were evaluated for their best setup before the actual evaluation results were acquired using this setup. This parameter evaluation showed that the five different classes require different tuning of the parameters to achieve the best possible results. This shows that those parameters should be evaluated and tuned differently for every scenario the *CENFA* algorithm is used in. However, to gain comparable evaluation results, the tuning parameters for the five different classes were chosen similarly. The values used for the parameters can be found below in Table III.

TABLE III
VALUES CHOSEN FOR THE TUNING PARAMETERS

Parameter	Chosen Value
Dividing Factor	0.70
Confidence Value	0.70
Number of Bagged SVMs	10

CENFA and the three different classifiers used for comparison were evaluated considering different corpus sizes. Besides the 100% corpus with 10,300 job offers, also 75%, 50%, 25% and 10% corpus sizes were used. To present the results for accuracy evaluation in a compact way, 10% and 100% corpus size were chosen for presentation in this paper only. These extreme values were chosen to show on the one hand the feasibility of the approach with a small training set only and on the other hand the increasing accuracy for a large data set. The overall trend was similar for all corpus sizes and the accuracy values were steadily increasing with increasing corpus size for all approaches presented.

[3]http://www.cs.waikato.ac.nz/ml/weka/

When using 10% of the evaluation corpus for evaluation, on average 14,5% of the instances were declared as ambiguous by the base classifier. Using the full evaluation corpus, 4.47% were declared as ambiguous. This trend is natural since the base classifier becomes more robust with a higher number of training instances.

In what follows we highlight different aspects of our evaluation.

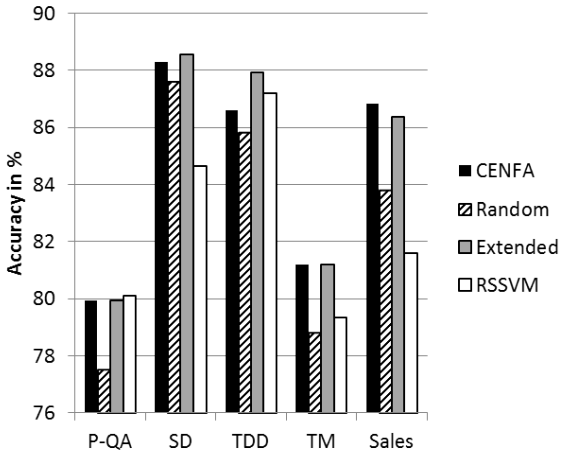


Fig. 2. The accuracy at 10% corpus size

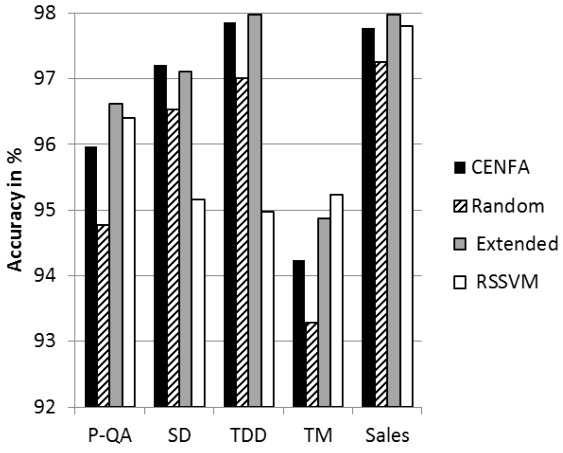
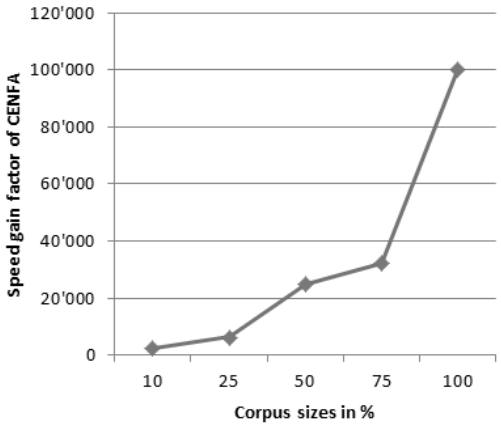
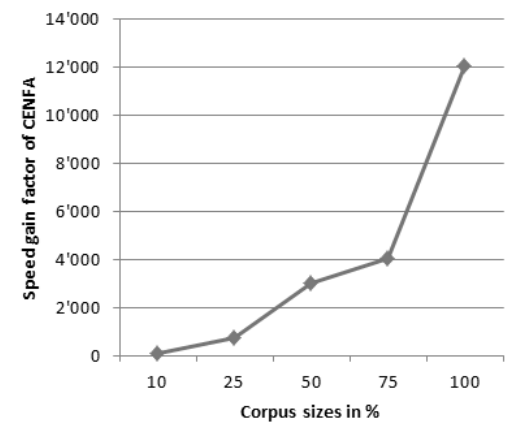


Fig. 3. The accuracy at 100% corpus size

1) Overall Accuracy: Figure 2 and Figure 3 show the performances of all the classifiers. *CENFA* can be seen to



(a) Speed gain factor of CENFA compared to RSSVM



(b) Speed gain factor of CENFA compared to Extended

Fig. 4. Compared time performance of different classifiers for different corpus sizes

perform very differently for the different classes at 10% corpus size in Figure 2. It ranges from 79.94% accuracy for the P-QA class up to 88.27% for the SD class, reaching an average of 84.57% with a standard deviation of 3.74%. This variation is likely due to the generally chosen tuning parameters which are fixed for the five classes instead of being tuned individually. Figure 3 shows that this variation decreases for bigger training corpora because the classifiers are trained thoroughly and reach towards the upper boundary of 100% accuracy. Here, *CENFA* can be seen to achieve between 94.25% and 97.86% accuracy, reaching an average of 96.61% and a standard deviation of 1.52%. Interestingly, the accuracy along the classes does not correspond with the number of positive examples per class.

2) Active Learning Accuracy: Figure 3 shows that the *CENFA* classifier was able to outperform the *Random* classifier at a corpus size of 100% in every class by between 1.2% and 0.5%. Figure 2 shows it outperforms the *Random* classifier at a corpus size of 10% by between 0.7% and 3.0%. By applying *active learning*, *CENFA* reaches higher accuracy for the same amount of training documents. By comparing Figure 2 and Figure 3 one can see that a larger number of training documents results in a better accuracy. Combining these two observations this means on the other hand that the *CENFA* classifier can reach the same accuracy by using fewer training documents compared to the *Random* classifier and therefore is more efficient.

3) Ensemble Learning Accuracy and Computational Complexity: As shown in Figure 3, the *CENFA* classifier was able to outperform the *RSSVM* classifier at a corpus size of 100% for two classes by 2.1% and 2.9% and reaches the same accuracy for one class. The *RSSVM* classifier on the other hand outperforms the *CENFA* for two classes by 0.4% and 1.0%. Figure 2 shows that *CENFA* outperforms the *RSSVM* at a corpus size of 10% for three classes by between 1.9% and 5.2% while the *RSSVM* is more accurate for two classes by 0.2% and 0.6%.

The differences of the results between the classes is due to the fixed tuning parameters which causes *CENFA's* performance to vary for the different classes while the *RSSVM* is only influenced in terms of numbers of training documents by those parameters. On average, the *CENFA* classifier reaches higher accuracy and can therefore be regarded as more effective.

However, another interesting evaluation parameter besides the accuracy is the build time. For the 100% corpus size the *CENFA* base classifier took 4,976.41 seconds to train and the special classifier took 0.05 seconds to re-train on average. The *RSSVM* took about 613.31 seconds to train. This means the *RSSVM* is about 8 times faster on the first training but *CENFA* is up to 12,000 times faster on every re-train iteration which makes it more efficient in the long run. Additionally, the speed gain factor of the *CENFA* compared to the *RSSVM* increases from small corpora to larger corpora which can be seen in Fig. 4a.

4) Uncompromising Accuracy and Computational Complexity: Figure 3 shows that the *Extended* classifier outperforms the *CENFA* classifier at a corpus size of 100% in all but the *SD* class. However, the maximal improvement of the *Extended* is at 0.6% and the average is at 0.15%. At a corpus size of 10% the *CENFA* reaches the same accuracy as the *Extended* for two classes, is outperformed for two classes by 0.3% and 1.3%, and reaches a higher accuracy for the Sales class by 0.5%. The average accuracy loss of the *CENFA* against the *Extended* is at 0.2%. That means that the *CENFA* classifier almost retains the *Extended* classifier's efficacy.

Again the build time of the classifier at a corpus size of 100% is also considered. Compared to a re-train build time of 0.05 seconds of *CENFA*, the *Extended* took 5,106.64 seconds. This means *CENFA* is up to 100,000 times faster and thus proves to be more efficient. Also, the speed gain factor of *CENFA* compared to the *Extended* increases from small corpora to larger corpora which can be seen in Figure 4b.

V. Conclusion and Future Work

The evaluation shows that the CENFA learner provides a combination of the strengths of ensemble and active learning. It is able to increase efficacy and efficiency compared to pure ensemble and active learning respectively. Compared to a standard combination of ensemble and active learning it almost retains the effectiveness and increases the efficiency substantially. In terms of time, the CENFA is up to 100.000 times faster.

The provided solution of the CENFA learner was created to classify Web documents and especially job offers. The approach needs to be evaluated in other domains of Web documents. It would also be interesting to apply this method in different classification scenarios, where entities other than Web documents have to be classified. The concept is independent from the underlying algorithm used (SVM), hence different algorithms can be tested in such an environment. Even base and specialized classifier could be applied using different algorithms. CENFA was evaluated with pre-annotated training sets simulating human feedback. Applying the algorithm in an actual active learning environment is a required step of evaluation in order to prove its suitability in a real-world scenario.

Acknowledgements

The work presented in this paper was partly funded by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IS12054 and partially funded in the framework of Hessen Modell Projekte, financed with funds of LOEWE-State Offensive for the Development of Scientific and Economic Excellence (HA project no. 292/11-37). The responsibility for the contents of this publication lies with the authors. We thank kimeta GmbH for the essential help assisting with building the evaluation corpus.

References

[1] Netcraft, "November 2013 web server survey," http://news.netcraft.com/archives/2013/11/01/november-2013-web-server-survey.html, year 2013, [Online; accessed 18-November-2013].

[2] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. Cambridge University Press Cambridge, 2008, vol. 1.

[3] G. Salton and C. Buckley, "Term weighting approaches in automatic text retrieval," *Information Processing Management*, vol. 24, no. 5, pp. 513–523, 1988.

[4] T. Joachims, "A statistical learning learning model of text classification for support vector machines," in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001, pp. 128–136. [Online]. Available: http://dl.acm.org/citation.cfm?id=383974

[5] N. Tripathi, M. Oakes, and S. Wermter, "A fast subspace text categorization method using parallel classifiers," in *Computational Linguistics and Intelligent Text Processing*. Springer, 2012, pp. 132–143. [Online]. Available: http://link.springer.com/chapter/10.1007/978-3-642-28601-8_12

[6] F. Fukumoto, Y. Suzuki, and S. Matsuyoshi, "Text classification from positive and unlabeled data using misclassified data correction," in *Proceedings of the the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, 2013, pp. 474–478.

[7] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2011.

[8] C. C. Aggarwal, *Mining text data*. Springer, 2012.

[9] B. Settles, M. Craven, and L. Friedland, "Active learning with real annotation costs," in *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, 2008, pp. 1–10. [Online]. Available: http://dl.acm.org/citation.cfm?id=1557119

[10] Y. Fu, X. Zhu, and B. Li, "A survey on instance selection for active learning," *Knowledge and Information Systems*, vol. 35, no. 2, pp. 249–283, May 2013. [Online]. Available: http://link.springer.com/article/10.1007/s10115-012-0507-8

[11] B. Yang, J.-T. Sun, T. Wang, and Z. Chen, "Effective multi-label active learning for text classification," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '09. New York, NY, USA: ACM, 2009, pp. 917–926. [Online]. Available: http://doi.acm.org/10.1145/1557019.1557119

[12] B. Settles, "Active learning literature survey," *University of Wisconsin on Active Learning, Madison*, 2010.

[13] J. Zhu and M. Ma, "Uncertainty-based active learning with instability estimation for text classification," *ACM Trans. Speech Lang. Process.*, vol. 8, no. 4, pp. 5:1–5:21, Feb. 2012. [Online]. Available: http://doi.acm.org/10.1145/2093153.2093154

[14] X. Li and C. G. Snoek, "Classifying tag relevance with relevant positive and negative examples," in *Proceedings of the 21st ACM International Conference on Multimedia*, ser. MM '13. New York, NY, USA: ACM, 2013, pp. 485–488. [Online]. Available: http://doi.acm.org/10.1145/2502081.2502129

[15] S. Schnitzer, "Effective classification of ambiguous web documents incorporating human feedback efficiently," Master's thesis, University of Applied Sciences Darmstadt, Faculty of Computer Science, Darmstadt, Germany, 2013.

[16] J. Platt, "Fast training of support vector machines using sequential minimal optimization," in *Advances in Kernel Methods - Support Vector Learning*, B. Schoelkopf, C. Burges, and A. Smola, Eds. MIT Press, 1998. [Online]. Available: http://dl.acm.org/citation.cfm?id=299105