Evaluating the Impact of Coder Errors on Active Learning

Ines Rehbein
Computational Linguistics
Saarland University
rehbein@coli.uni-sb.de

Josef Ruppenhofer
Computational Linguistics
Saarland University
josefr@coli.uni-sb.de

Abstract

Active Learning (AL) has been proposed as a technique to reduce the amount of annotated data needed in the context of supervised classification. While various simulation studies for a number of NLP tasks have shown that AL works well on goldstandard data, there is some doubt whether the approach can be successful when applied to noisy, real-world data sets. This paper presents a thorough evaluation of the impact of annotation noise on AL and shows that systematic noise resulting from biased coder decisions can seriously harm the AL process. We present a method to filter out inconsistent annotations during AL and show that this makes AL far more robust when applied to noisy data.

1 Introduction

Supervised machine learning techniques are still the mainstay for many NLP tasks. There is, however, a well-known bottleneck for these approaches: the amount of high-quality data needed for training, mostly obtained by human annotation. Active Learning (AL) has been proposed as a promising approach to reduce the amount of time and cost for human annotation. The idea behind active learning is quite intuitive: instead of annotating a large number of randomly picked instances we carefully select a small number of instances that are maximally informative for the machine learning classifier. Thus a smaller set of data points is able to boost classifier performance and to yield an accuracy comparable to the one obtained when training the same system on a larger set of randomly chosen data.

Active learning has been applied to several NLP tasks like part-of-speech tagging (Ringger et al., 2007), chunking (Ngai and Yarowsky, 2000), syntactic parsing (Osborne and Baldridge, 2004; Hwa, 2004), Named Entity Recognition (Shen et al., 2004; Laws and Schütze, 2008; Tomanek and Hahn, 2009), Word Sense Disambiguation (Chen et al., 2006; Zhu and Hovy, 2007; Chan and Ng, 2007), text classification (Tong and Koller, 1998) or statistical machine translation (Haffari and Sarkar, 2009), and has been shown to reduce the amount of annotated data needed to achieve a certain classifier performance, sometimes by as much as half. Most of these studies, however, have only simulated the active learning process using goldstandard data. This setting is crucially different from a real world scenario where we have to deal with erroneous data and inconsistent annotation decisions made by the human annotators. While simulations are an indispensable instrument to test different parameters and settings, it has been shown that when applying AL to highly ambiguous tasks like e.g. Word Sense Disambiguation (WSD) with fine-grained sense distinctions, AL can actually harm the learning process (Dang, 2004; Rehbein et al., 2010). Dang suggests that the lack of a positive effect of AL might be due to inconsistencies in the human annotations and that AL cannot efficiently be applied to tasks which need double blind annotation with adjudication to insure a sufficient data quality. Even if we take a more optimistic view and assume that AL might still be useful even for tasks featuring a high degree of ambiguity, it remains crucial to address the problem of annotation noise and its impact on AL.

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pages 43–51,
Portland, Oregon, June 19-24, 2011.

In this paper we present a thorough evaluation of the impact of annotation noise on AL. We simulate different types of coder errors and assess the effect on the learning process. We propose a method to detect inconsistencies and remove them from the training data, and show that our method does alleviate the problem of annotation noise in our experiments.

The paper is structured as follows. Section 2 reports on recent research on the impact of annotation noise in the context of supervised classification. Section 3 describes the experimental setup of our simulation study and presents results. In Section 4 we present our filtering approach and show its impact on AL performance. Section 5 concludes and outlines future work.

2 Related Work

We are interested in the question whether or not AL can be successfully applied to a supervised classification task where we have to deal with a considerable amount of inconsistencies and noise in the data, which is the case for many NLP tasks (e.g. sentiment analysis, the detection of metaphors, WSD with fine-grained word senses, to name but a few). Therefore we do not consider part-of-speech tagging or syntactic parsing, where coders are expected to agree on most annotation decisions. Instead, we focus on work on AL for WSD, where inter-coder agreement (at least for fine-grained annotation schemes) usually is much lower than for the former tasks.

2.1 Annotation Noise

Studies on active learning for WSD have been limited to running simulations of AL using gold standard data and a coarse-grained annotation scheme (Chen et al., 2006; Chan and Ng, 2007; Zhu and Hovy, 2007). Two exceptions are Dang (2004) and Rehbein et al. (2010) who both were not able to replicate the positive findings obtained for AL for WSD on coarse-grained sense distinctions. A possible reason for this failure is the amount of annotation noise in the training data which might mislead the classifier during the AL process. Recent work on the impact of annotation noise on a machine learning task (Reidsma and Carletta, 2008) has shown that random noise can be tolerated in supervised learning, while systematic errors (as caused by biased annotators) can seriously impair the performance of a supervised classifier even if the observed accuracy of the classifier on a test set coming from the same population as the training data is as high as 0.8.

Related work (Beigman Klebanov et al., 2008; Beigman Klebanov and Beigman, 2009) has been studying annotation noise in a multi-annotator setting, distinguishing between *hard* cases (unreliably annotated due to genuine ambiguity) and *easy* cases (reliably annotated data). The authors argue that even for those data points where the annotators agreed on one particular class, a proportion of the agreement might be merely due to chance. Following this assumption, the authors propose a measure to estimate the amount of annotation noise in the data after removing all hard cases. Klebanov et al. (2008; 2009) show that, according to their model, high inter-annotator agreement (κ) achieved in an annotation scenario with two annotators is no guarantee for a high-quality data set. Their model, however, assumes that a) all instances where annotators disagreed are in fact hard cases, and b) that for the hard cases the annotators decisions are obtained by coin-flips. In our experience, some amount of disagreement can also be observed for easy cases, caused by attention slips or by a deviant interpretation of some class(es) by one of the annotators, and the annotation decision of an individual annotator cannot so much be described as random choice (coin-flip) but as systematically biased selection, causing the types of errors which have been shown to be problematic for supervised classification (Reidsma and Carletta, 2008).

Further problems arise in the AL scenario where the instances to be annotated are selected as a function of the sampling method and the annotation judgements made before. Therefore, Beigman and Klebanov Beigman (2009)'s approach of identifying unreliably annotated instances by disagreement is not applicable to AL, as most instances are annotated only once.

2.2 Annotation Noise and Active Learning

For AL to be succesful, we need to remove systematic noise in the training data. The challenge we face is that we only have a small set of seed data and no information about the reliability of the annotations

assigned by the human coders.

Zhu et al. (2008) present a method for detecting outliers in the pool of unannotated data to prevent these instances from becoming part of the training data. This approach is different from ours, where we focus on detecting annotation noise in the manually labelled training data produced by the human coders.

Schein and Ungar (2007) provide a systematic investigation of 8 different sampling methods for AL and their ability to handle different types of noise in the data. The types of noise investigated are a) prediction residual error (the portion of squared error that is independent of training set size), and b) different levels of confusion among the categories. Type a) models the presence of unknown features that influence the true probabilities of an outcome: a form of noise that will increase residual error. Type b) models categories in the data set which are intrinsically hard to disambiguate, while others are not. Therefore, type b) errors are of greater interest to us, as it is safe to assume that intrinsically ambiguous categories will lead to biased coder decisions and result in the systematic annotation noise we are interested in.

Schein and Ungar observe that none of the 8 sampling methods investigated in their experiment achieved a significant improvement over the random sampling baseline on type b) errors. In fact, entropy sampling and margin sampling even showed a decrease in performance compared to random sampling. For AL to work well on noisy data, we need to identify and remove this type of annotation noise during the AL process. To the best of our knowledge, there is no work on detecting and removing annotation noise by human coders during AL.

3 Experimental Setup

To make sure that the data we use in our simulation is as close to real-world data as possible, we do not create an artificial data set as done in (Schein and Ungar, 2007; Reidsma and Carletta, 2008) but use real data from a WSD task for the German verb *drohen* (threaten).[1] *Drohen* has three different word senses which can be disambiguated by humans with a high accuracy.[2] This point is crucial to our setup. To control the amount of noise in the data, we need to be sure that the initial data set is noise-free.

For classification we use a maximum entropy classifier.[3] Our sampling method is uncertainty sampling (Lewis and Gale, 1994), a standard sampling heuristic for AL where new instances are selected based on the confidence of the classifier for predicting the appropriate label. As a measure of uncertainty we use Shannon entropy (1) (Zhang and Chen, 2002) and the *margin* metric (2) (Schein and Ungar, 2007). The first measure considers the model's predictions q for each class c and selects those instances from the pool where the Shannon entropy is highest.

$$-\sum_{c} q_c \, log \, q_c \tag{1}$$

The second measure looks at the difference between the largest two values in the prediciton vector q, namely the two predicted classes c, c' which are, according to our model, the most likely ones for instance x_n, and selects those instances where the difference (*margin*) between the two predicted probabilities is the smallest. We discuss some details of this metric in Section 4.

$$M_n = |P(c|x_n) - P(c'|x_n)| \tag{2}$$

The features we use for WSD are a combination of context features (word token with window size 11 and POS context with window size 7), syntactic features based on the output of a dependency parser[4] and semantic features based on GermaNet hyperonyms. These settings were tuned to the target verb by (Rehbein et al., 2009). All results reported below are averages over a 5-fold cross validation.

3.1 Simulating Coder Errors in AL

Before starting the AL trials we automatically separate the 2,500 sentences into test set (498 sentences) and pool (2,002 sentences),[5] retaining the overall distribution of word senses in the data set. We insert a varying amount of noise into the pool data,

[1] The data has been provided by the SALSA project: `http://www.coli.uni-saarland.de/projects/salsa`

[2] In a pilot study where two human coders assigned labels to a set of 100 sentences, the coders agreed on 99% of the data.

[3] `http://maxent.sourceforge.net`

[4] The MaltParser: `http://maltparser.org`

[5] The split has been made automatically, the unusual numbers are caused by rounding errors.

	test	pool		
% errors	**0%**	**0%**	**ALrand 30%**	**ALbias 30%**
drohen1-salsa	126	506	524	514
Comittment	129	520	522	327
Run_risk	243	976	956	1161
Total	498	2002	2002	2002

Table 1: Distribution of word senses in pool and test sets

starting from 0% up to 30% of noise, increasing by 2% in each trial.

We assess the impact of annotation noise on active learning in three different settings. In the first setting, we randomly select new instances from the pool (random sampling; *rand*). In the second setting, we randomly replace n percent of all labels (from 0 to 30) in the pool by another label before starting the active learning trial, but retain the distribution of the different labels in the pool data (active learning with random errors); (Table 1, *ALrand*, 30%). In the third setting we simulate biased decisions by a human annotator. For a certain fraction (0 to 30%) of instances of a particular non-majority class, we substitute the majority class label for the gold label, thereby producing a more skewed distribution than in the original pool (active learning with biased errors); (Table 1, *ALbias*, 30%).

For all three settings (*rand, ALrand, ALbias*) and each degree of noise (0-30%), we run active learning simulations on the already annotated data, simulating the annotation process by selecting one new, prelabelled instance per trial from the pool and, instead of handing them over to a human coder, assigning the known (possibly erroneous) label to the instance and adding it to the training set. We use the same split (test, pool) for all three settings and all degrees of noise, with identical test sets for all trials.

3.2 Results

Figure 1 shows active learning curves for the different settings and varying degrees of noise. The horizontal black line slightly below 0.5 accuracy shows the majority baseline (the performance obtained when always assigning the majority class). For all degrees of randomly inserted noise, active learning (*ALrand*) outperforms random sampling (*rand*) at an early stage in the learning process. Looking at the biased errors (*ALbias*), we see a different picture. With a low degree of noise, the curves for *ALrand* and *ALbias* are very similar. When inserting more noise, performance for *ALbias* decreases, and with around 20% of biased errors in the pool AL performs worse than our random sampling baseline. In the random noise setting (*ALrand*), even after inserting 30% of errors AL clearly outperforms random sampling. Increasing the size of the seed data reduces the effect slightly, but does not prevent it (not shown here due to space limitations). This confirms the findings that under certain circumstances AL performs worse than random sampling (Dang, 2004; Schein and Ungar, 2007; Rehbein et al., 2010). We could also confirm Schein and Ungar (2007)'s observation that margin sampling is less sensitive to certain types of noise than entropy sampling (Table 2). Because of space limitations we only show curves for margin sampling. For entropy sampling, the general trend is the same, with results being slightly lower than for margin sampling.

4 Detecting Annotation Noise

Uncertainty sampling using the margin metric selects instances for which the difference between classifier predictions for the two most probable classes c, c' is very small (Section 3, Equation 2). When selecting unlabelled instances from the pool, this metric picks examples which represent regions of uncertainty between classes which have yet to be learned by the classifier and thus will advance the learning process. Our human coder, however, is not the perfect oracle assumed in most AL simulations, and might also assign incorrect labels. The filter approach has two objectives: a) to detect incorrect labels assigned by human coders, and b) to prevent the *hard* cases (following the terminology of Klebanov et al. (2008)) from becoming part of the training data.

We proceed as follows. Our approach makes use of the limited set of seed data S and uses heuristics to detect unreliably annotated instances. We assume that the instances in S have been validated thoroughly. We train an ensemble of classifiers E on subsets of S, and use E to decide whether or not a newly annotated instance should be added to the seed.

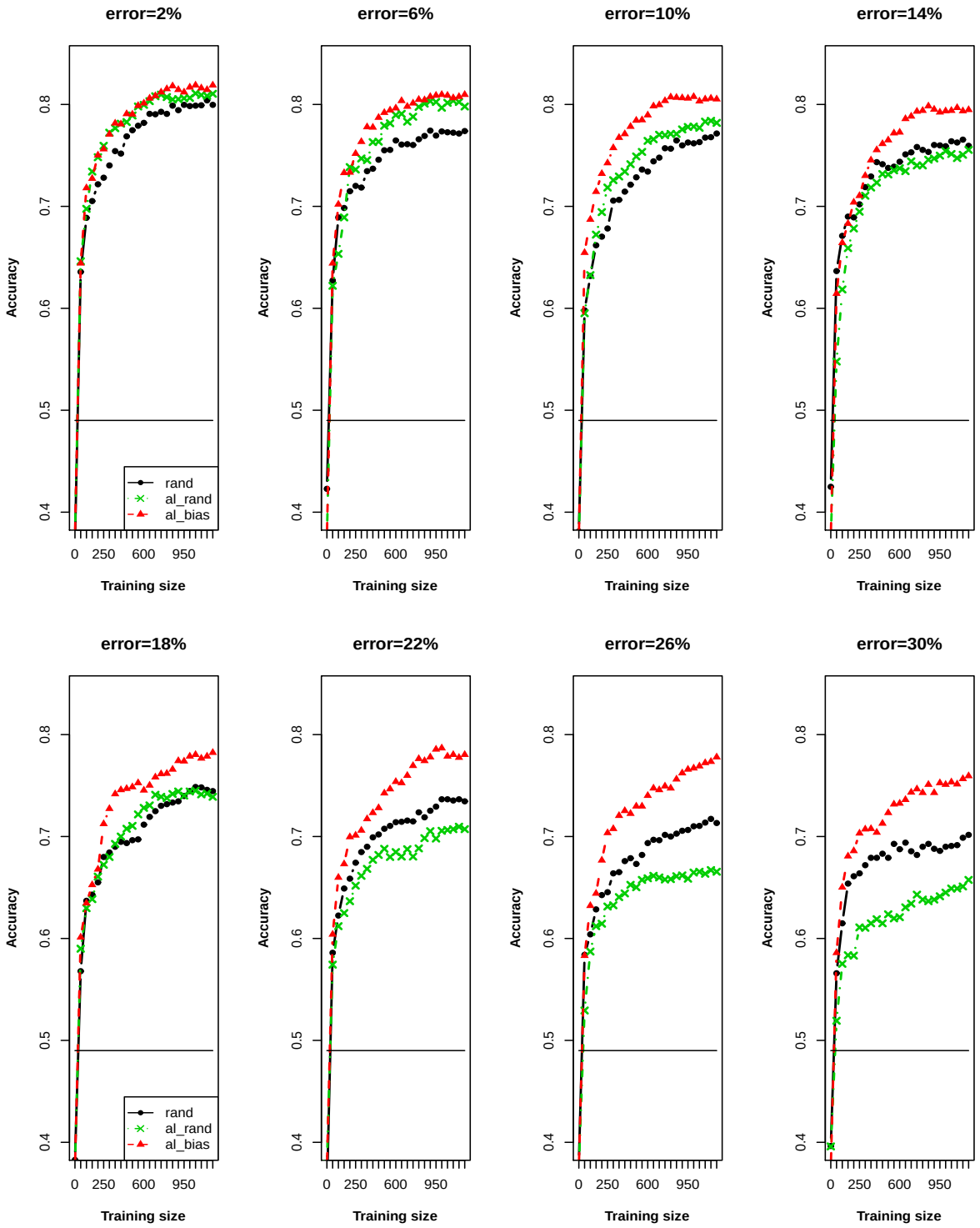

Figure 1: Active learning curves for varying degrees of noise, starting from 0% up to 30% for a training size up to 1200 instances (solid circle (black): random sampling; filled triangle point-up (red): AL with random errors; cross (green): AL with biased errors)

	filter	% error	0	4	8	12	16	20	24	28	30
	-	**rand**	0.763	0.752	0.736	0.741	0.726	0.708	0.707	0.677	0.678
entropy	-	**ALrand**	0.806	0.786	0.779	0.743	0.752	0.762	0.731	0.724	0.729
entropy	y	**ALrand**	0.792	0.786	0.777	0.760	0.771	0.748	0.730	0.729	0.727
margin	-	**ALrand**	0.795	0.795	0.782	0.771	0.758	0.755	0.737	0.719	0.708
margin	y	**ALrand**	0.800	0.785	0.773	0.777	0.765	0.766	0.734	0.735	0.718
entropy	-	**ALbias**	0.806	0.793	0.759	0.748	0.702	0.651	0.625	0.630	0.622
entropy	y	**ALbias**	0.802	0.781	0.777	0.735	0.702	0.678	0.687	0.624	0.616
margin	-	**ALbias**	0.795	0.789	0.770	0.753	0.706	0.684	0.656	0.634	0.624
margin	y	**ALbias**	0.787	0.781	0.787	0.768	0.739	0.700	0.671	0.653	0.651

Table 2: Accuracy for the different sampling methods without and with filtering after adding 500 instances to the seed data

There are a number of problems with this approach. First, there is the risk of overfitting S. Second, we know that classifier accuracy in the early phase of AL is low. Therefore, using classifier predictions at this stage to accept or reject new instances could result in poor choices that might harm the learning proceess. To avoid this and to generalise over S to prevent overfitting, we do not directly train our ensemble on instances from S. Instead, we create new feature vectors F_{gen} on the basis of the feature vectors F_{seed} in S. For each class in S, we extract all attribute-value pairs from the feature vectors for this particular class. For each class, we randomly select features (with replacement) from F_{seed} and combine them into a new feature vector F_{gen}, retaining the distribution of the different classes in the data. As a result, we obtain a more general set of feature vectors F_{gen} with characteristic features being distributed more evenly over the different feature vectors.

In the next step we train $n = 5$ maximum entropy classifiers on subsets of F_{gen}, excluding the instances last annotated by the oracle. Each subset is half the size of the current S. We use the ensemble to predict the labels for the new instances and, based on the predictions, accept or reject these, following the two heuristics below (also see Figure 2).

1. If all n ensemble classifiers agree on one label but disagree with the oracle $\Rightarrow$ reject.

2. If the sum of the margins predicted by the ensemble classifiers is below a particular theshold $t_{margin} \Rightarrow$ reject.

The threshold t_{margin} was set to 0.01, based on a qualitative data analysis.

AL with Filtering:

Input: annotated seed data S,
unannotated pool P
AL loop:

- train classifier C on S
- let C predict labels for data in P
- select new instances from P according to sampling method, hand over to oracle for annotation

Repeat: after every c new instances annotated by the oracle

- for each class in S, extract sets of features F_{seed}
- create new, more general feature vectors F_{gen} from this set (with replacement)
- train an ensemble E of n classifiers on different subsets of F_{gen}

  Filtering Heuristics:

  - **if** all n classifier in E agree on label but disagree with oracle:
    $\Rightarrow$ *remove instance from seed*
  - **if** margin is less than threshold t_{margin}:
    $\Rightarrow$ *remove instance from seed*

Until done

Figure 2: Heuristics for filtering unreliable data points (parameters used: initial seed size: 9 sentences, $c = 10$, $n = 5$, $t_{margin} = 0.01$)

In each iteration of the AL process, one new instance is selected using margin sampling. The instance is presented to the oracle who assigns a label. Then the instance is added to the seed data, thus influencing the selection of the next data point to be annotated. After 10 new instances have been added, we apply the filter technique which finally decides whether the newly added instances will remain in the seed data or will be removed.

Figure 3 shows learning curves for the filter approach. With increasing amount of errors in the pool, a clear pattern emerges. For both sampling methods (ALrand, ALbias), the filtering step clearly improves results. Even for the noisier data sets with up to 26% of errors, ALbias with filtering performs at least as well as random sampling.

4.1 Error Analysis

Next we want to find out what kind of errors the system could detect. We want to know whether the approach is able to detect the errors previously inserted into the data, and whether it manages to identify hard cases representing true ambiguities.

To answer these questions we look at one fold of the ALbias data with 10% of noise. In 1,200 AL iterations the system rejected 116 instances (Table 3). The major part of the rejections was due to the majority vote of the ensemble classifiers (first heuristic, H1) which rejects all instances where the ensemble classifiers agree with each other but disagree with the human judgement. Out of the 105 instances rejected by H1, 41 were labelled incorrectly. This means that we were able to detect around half of the incorrect labels inserted in the pool.

11 instances were filtered out by the margin threshold (H2). None of these contained an incorrect label. On first glance H2 seems to be more lenient than H1, considering the number of rejected sentences. This, however, could also be an effect of the order in which we apply the filters.

errors inserted in pool	173
err. instances selected by AL	93
instances rejected by H1+H2	116
instances rejected by H1	105
true errors rejected by H1	41
instances rejected by H2	11
true errors rejected by H2	0

Table 3: Error analysis of the instances rejected by the filtering approach

The different word senses are evenly distributed over the rejected instances (H1: Commitment 30, drohen1-salsa 38, Run_risk 36; H2: Commitment 3, drohen1-salsa 4, Run_risk 4). This shows that there is less uncertainty about the majority word sense, Run_risk.

It is hard to decide whether the correctly labelled instances rejected by the filtering method would have helped or hurt the learning process. Simply adding them to the seed data after the conclusion of AL would not answer this question, as it would merely tell us whether they improve classification accuracy further, but we still would not know what impact these instances would have had on the selection of instances during the AL process.

5 Conclusions

This paper shows that certain types of annotation noise cause serious problems for active learning approaches. We showed how biased coder decisions can result in an accuracy for AL approaches which is below the one for random sampling. In this case, it is necessary to apply an additional filtering step to remove the noisy data from the training set. We presented an approach based on a resampling of the features in the seed data and guided by an ensemble of classifiers trained on the resampled feature vectors. We showed that our approach is able to detect a certain amount of noise in the data.

Future work should focus on finding optimal parameter settings to make the filtering method more robust even for noisier data sets. We also plan to improve the filtering heuristics and to explore further ways of detecting human coder errors. Finally, we plan to test our method in a real-world annotation scenario.

6 Acknowledgments

This work was funded by the German Research Foundation DFG (grant PI 154/9-3). We would like to thank the anonymous reviewers for their helpful comments and suggestions.

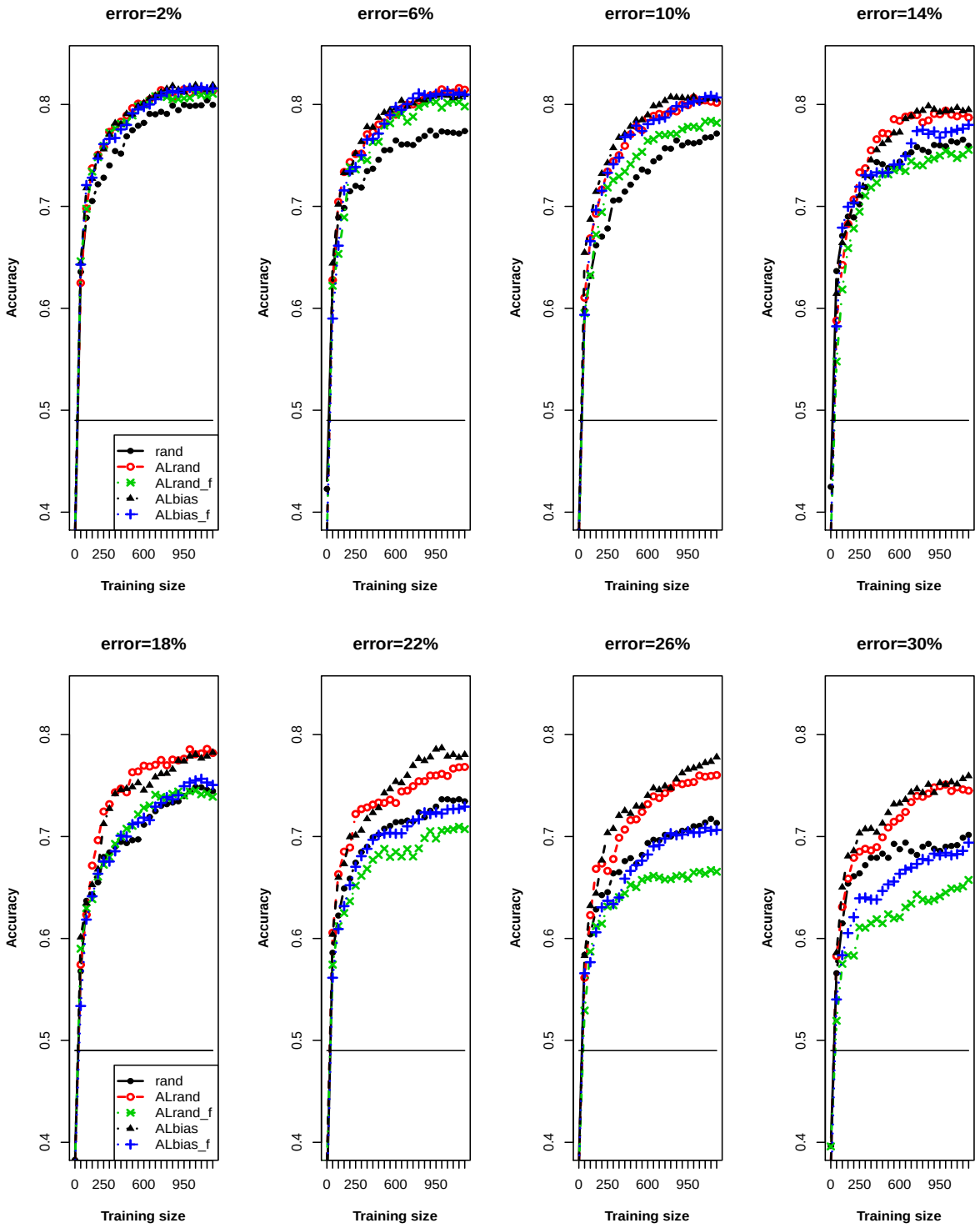

Figure 3: Active learning curves for varying degrees of noise, starting from 0% up to 30% for a training size up to 1200 instances (solid circle (black): random sampling; open circle (red): ALrand; cross (green): ALrand with filtering; filled triangle point-up (black): ALbias; plus (blue): ALbias with filtering)

References

Beata Beigman Klebanov and Eyal Beigman. 2009. From annotator agreement to noise models. *Computational Linguistics*, 35:495–503, December.

Beata Beigman Klebanov, Eyal Beigman, and Daniel Diermeier. 2008. Analyzing disagreements. In *Proceedings of the Workshop on Human Judgements in Computational Linguistics*, HumanJudge '08, pages 2–7, Morristown, NJ, USA. Association for Computational Linguistics.

Yee Seng Chan and Hwee Tou Ng. 2007. Domain adaptation with active learning for word sense disambiguation. In *Proceedings of ACL-2007*.

Jinying Chen, Andrew Schein, Lyle Ungar, and Martha Palmer. 2006. An empirical study of the behavior of active learning for word sense disambiguation. In *Proceedings of NAACL-2006*, New York, NY.

Hoa Trang Dang. 2004. *Investigations into the role of lexical semantics in word sense disambiguation*. PhD dissertation, University of Pennsylvania, Pennsylvania, PA.

Gholamreza Haffari and Anoop Sarkar. 2009. Active learning for multilingual statistical machine translation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, pages 181–189. Association for Computational Linguistics.

Rebecca Hwa. 2004. Sample selection for statistical parsing. *Computational Linguistics*, 30(3):253–276.

Florian Laws and H. Schütze. 2008. Stopping criteria for active learning of named entity recognition. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, Manchester, UK, August.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of ACM-SIGIR*, Dublin, Ireland.

Grace Ngai and David Yarowsky. 2000. Rule writing or annotation: cost-efficient resource usage for base noun phrase chunking. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 117–125, Stroudsburg, PA, USA. Association for Computational Linguistics.

Miles Osborne and Jason Baldridge. 2004. Ensemble-based active learning for parse selection. In *Proceedings of HLT-NAACL 2004*.

Ines Rehbein, Josef Ruppenhofer, and Jonas Sunde. 2009. Majo - a toolkit for supervised word sense disambiguation and active learning. In *Proceedings of the 8th Workshop on Treebanks and Linguistic Theories (TLT-8)*, Milano, Italy.

Ines Rehbein, Josef Ruppenhofer, and Alexis Palmer. 2010. Bringing active learning to life. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, Beijing, China.

Dennis Reidsma and Jean Carletta. 2008. Reliability measurement without limits. *Computational Linguistics*, 34:319–326.

Eric Ringger, Peter McClanahan, Robbie Haertel, George Busby, Marc Carmen, James Carroll, Kevin Seppi, and Deryle Lonsdale. 2007. Active learning for part-of-speech tagging: Accelerating corpus annotation. In *Proceedings of the Linguistic Annotation Workshop*, Prague.

Andrew I. Schein and Lyle H. Ungar. 2007. Active learning for logistic regression: an evaluation. *Machine Learning*, 68:235–265.

Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew-Lim Tan. 2004. Multi-criteria-based active learning for named entity recognition. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Katrin Tomanek and Udo Hahn. 2009. Reducing class imbalance during active learning for named entity annotation. In *Proceedings of the 5th International Conference on Knowledge Capture*, Redondo Beach, CA.

Simon Tong and Daphne Koller. 1998. Support vector machine active learning with applications to text classification. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML-00)*, pages 287–295.

Cha Zhang and Tsuhan Chen. 2002. An active learning framework for content-based information retrieval. *IEEE Transactions on Multimedia*, 4(2):260–268.

Jingbo Zhu and Edward Hovy. 2007. Active learning for word sense disambiguation with methods for addressing the class imbalance problem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Prague, Czech Republic.

Jingbo Zhu, Huizhen Wang, Tianshun Yao, and Benjamin K. Tsou. 2008. Active learning with sampling by uncertainty and density for word sense disambiguation and text classification. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, Manchester, UK.