NAACL HLT 2009

Active Learning for Natural Language Processing (ALNLP-09)

Proceedings of the Workshop

June 5, 2009
Boulder, Colorado

Production and Manufacturing by
Omnipress Inc.
2600 Anderson Street
Madison, WI 53707
USA

Endorsed by the following ACL Special Interest Groups:

- SIGNLL, Special Interest Group for Natural Language Learning
- SIGANN, Special Interest Group for Annotation

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
209 N. Eighth Street
Stroudsburg, PA 18360
USA
Tel: +1-570-476-8006
Fax: +1-570-476-0860
acl@aclweb.org

ISBN 978-1-932432-40-4

Introduction

Welcome to the workshop on Active Learning for Natural Language Processing!

We started organizing this workshop in mid-2008 after strong encouragement in response to some of our own work in the area. As we gathered members of the program committee, the timeliness of the topic resonated with several of them: the growing body of knowledge on active learning and on active learning for NLP in particular makes this topic one worth exploring in a focused workshop rather than in isolated papers in occasional, far-flung conferences.

Labeled data is a prerequisite for many popular algorithms in natural language processing and machine learning. While it is possible to obtain large amounts of annotated data for well-studied languages in well-studied domains and well-studied problems, labeled data are rarely available for less common languages, domains, or problems. Unfortunately, obtaining human annotations for linguistic data is labor-intensive and typically the costliest part of the acquisition of an annotated corpus. It has been shown before that active learning can be employed to reduce annotation costs but not at the expense of quality. While diverse work over the past decade has demonstrated the possible advantages of active learning for corpus annotation and NLP applications, active learning is not widely used in many ongoing data annotation tasks. Much of the machine learning literature on the topic has focused on active learning for classification problems with less attention devoted to the kinds of problems encountered in NLP. Related topics such as distributed "human computation", cost-sensitive machine learning, and semi-supervised learning of all kinds are also growing in number as we search for the best ways to overcome the data acquisition bottleneck.

We were interested in bringing together researchers to explore the challenges and opportunities of active learning for NLP tasks, language acquisition, and language learning, and we have been rewarded with excellent submissions and a promising program. The workshop received sixteen submissions, eight of which are included in the final program. Two of the accepted papers are short papers which address ongoing work and pertinent issues. We hope that this gathering and these proceedings begin to shed more light on active learning for NLP classification tasks, sequence labeling, parsing, semantics, and other more complex tasks. The papers in the program also begin to address issues involving the application of active learning in real annotation projects.

We are especially grateful to the diverse and helpful program committee, whose reviews were careful and thoughtful. We are also grateful to all of the researchers who submitted their work for consideration. For the record, more information about the workshop is available online at http://nlp.cs.byu.edu/alnlp/.

Best regards,

Eric Ringger, Robbie Haertel, and Katrin Tomanek

Organizers:

Eric Ringger, Brigham Young University (USA)
Robbie Hertel, Brigham Young University (USA)
Katrin Tomanek, University of Jena (Germany)

Program Committee:

Shlomo Argamon, Illinois Institute of Technology (USA)
Jason Baldridge, University of Texas at Austin (USA)
Markus Becker, SPSS (UK)
Ken Church, Microsoft Research (USA)
Hal Daume, University of Utah (USA)
Robbie Haertel, Brigham Young University (USA)
Ben Hachey, University of Edinburgh (UK)
Udo Hahn, University of Jena (Germany)
Eric Horvitz, Microsoft Research (USA)
Rebecca Hwa, University of Pittsburgh (USA)
Ashish Kapoor, Microsoft Research (USA)
Mark Liberman, University of Pennsylvania/LDC (USA)
Prem Melville, IBM T.J. Watson Research Center (USA)
Ray Mooney, University of Texas at Austin (USA)
Miles Osborne, University of Edinburgh (UK)
Eric Ringger, Brigham Young University (USA)
Kevin Seppi, Brigham Young University (USA)
Burr Settles, University of Wisconsin (USA)
Victor Sheng, New York University (USA)
Katrin Tomanek, University of Jena (Germany)
Jingbo Zhu, Northeastern University (China)

Invited Speakers:

Burr Settles, University of Wisconsin (USA)
Robbie Haertel, Brigham Young University (USA)

Table of Contents

Conference Program

Friday, June 5, 2009

8:30–9:00 Opening Remarks

9:00–10:00 Invited Talk by Burr Settles

Session 1: Anaphora Resolution

10:00–10:30 *Active Learning for Anaphora Resolution*
Caroline Gasperin

10:30–11:00 Break

Session 2: Multiple Annotators and Cost Considerations

11:00–11:30 *On Proper Unit Selection in Active Learning: Co-Selection Effects for Named Entity Recognition*
Katrin Tomanek, Florian Laws, Udo Hahn and Hinrich Schütze

11:30–12:00 *Estimating Annotation Cost for Active Learning in a Multi-Annotator Environment*
Shilpa Arora, Eric Nyberg and Carolyn P. Rosé

12:00–12:30 *Data Quality from Crowdsourcing: A Study of Annotation Selection Criteria*
Pei-Yun Hsueh, Prem Melville and Vikas Sindhwani

12:30–2:00 Lunch

Friday, June 5, 2009 (continued)

Session 3: Real Annotators and Experts

2:00–2:30 *Evaluating Automation Strategies in Language Documentation*
Alexis Palmer, Taesun Moon and Jason Baldridge

2:30–3:00 *A Web Survey on the Use of Active Learning to Support Annotation of Text Data*
Katrin Tomanek and Fredrik Olsson

3:00–3:30 Invited Talk by Robbie Haertel

3:30–4:00 Break

Session 4: New Methods

4:00–4:30 *Active Dual Supervision: Reducing the Cost of Annotating Examples and Features*
Prem Melville and Vikas Sindhwani

4:30–5:00 *Proactive Learning for Building Machine Translation Systems for Minority Languages*
Vamshi Ambati and Jaime Carbonell

5:00–5:30 Discussion

Active Learning for Anaphora Resolution

Caroline Gasperin
Computer Laboratory, University of Cambridge
15 JJ Thomson Avenue
Cambridge CB3 0FD, UK
cvg20@cl.cam.ac.uk

Abstract

In this paper we present our experiments with active learning to improve the performance of our probabilistic anaphora resolution system. We have adopted entropy-based uncertainty measures to select new instances to be added to our training data. The actively selected instances, however, were not more successful in improving the performance of the system than the same amount of randomly selected instances. The uncertainty measures we used behave differently from each other when selecting new instances, but none of them achieved remarkable performance. Further studies on active sample selection for anaphora resolution are necessary.

1 Introduction

Anaphora is the relation between two linguistic expressions in the discourse where the reader is referred back to the first of them when reading the second later in the text. Anaphora resolution can be understood as the process of identifying an anaphoric relation between two expressions in the text and consequently linking the two of them, one being the anaphor and the other being the antecedent. Manually annotating corpora with anaphoric links in order to use it as training or test data for a corpus-based anaphora resolution system is a particulary difficult and time consuming task, given the complex nature of the phenomenon.

We have developed a probabilistic model for resolution of non-pronominal anaphora and aim to improve its performance by acquiring incrementally and selectively more training data using active learning. We have adopted an uncertainty-based active learning approach in order to do that, and it uses our probabilistic model as the base classifier.

The uncertainty-based approach has been applied to, for instance, named-entity recognition by Shen et al. (2004) who report at least 80% reduction in annotation costs, parsing by Tang et al. (2002) who reports 67% savings, and parse selection by Baldridge and Osborne (2003) who report 60% savings. We are not aware of any work that has applied active learning to anaphora resolution.

For calculating the uncertainty of an anaphora resolution model, we feel the need to combine the information about the confindence of the model for the classification of each antecedent candidate associated to a given anaphor. We have tested three entropy-based uncertainty measures in order to select the instances to be added to the training data.

Our training corpus is composed of five full-length scientific articles from the biomedical domain. We have used this corpus to simulate active learning: we have divided our training data into two parts, one for the initial training and the other for active learning (simulating unlabelled data), and have compared the classifier performance when trained on a sample selected by active learning to its performance when trained on the same amount of randomly selected instances.

In the next section we describe our probabilistic model for anaphora resolution. In Section 3 we detail our training corpus. In Section **??** we describe the strategy we have adopted to select the samples to take part in the active learning, and in Section 5

Proceedings of the NAACL HLT Workshop on Active Learning for Natural Language Processing, pages 1–8,
Boulder, Colorado, June 2009.

we describe our experiments.

2 Anaphora resolution model

We have inplemented a probabilistic model for anaphora resolution in the biomedical domain (Gasperin and Briscoe, 2008). This model aims to resolve both coreferent and associative (also called bridging (Poesio and Vieira, 1998)) cases of non-pronominal anaphora. Table 1 shows examples of these types of anaphoric relations. Coreferent are the cases in which the anaphor and the antecedent refer to the same entity in the world, while associative cases are the ones in which the anaphor and antecedent refer to different but somehow related entities. We only take into account noun phrases referring to biomedical entities, since this was the focus of our resolution model. We consider two types of associative relations: biotype relations, which are anaphoric associative relations between noun phrases that share specific ontological relations in the biomedical domain; and set-member relations, in which the noun phrases share a set-membership relation. It is frequent however that some noun phrases do not have an antecedent, these are considered discourse-new cases, which we also aim to identify.

The probabilistic model results from a simple decomposition process applied to a conditional probability equation that involves several parameters (features). It is inspired by Ge et al.'s (1998) probabilistic model for pronoun resolution. The decomposition makes use of Bayes' theorem and independence assumptions, and aims to decrease the impact of data sparseness on the model, so that even small training corpora can be viable. The decomposed model can be thought of as a more sophisticated version of the naive-Bayes algorithm, since we consider the dependence among some of the features instead of full independence as in naive Bayes. Probabilistic models can return a confidence measure (probability) for each decision they make, which allow us to adopt techniques such as active learning for further processing.

Our model seeks to classify the relation between an anaphoric expression and an antecedent candidate as coreferent, biotype, set-member or neither. It computes the probability of each pair of anaphor and candidate for each class. The candidate with the highest overall probability for each class is selected as the antecedent for that class, or no antecedent is selected if the probability of no relation overcomes the positive probabilities; in this case, the expression is considered to be new to the discourse.

We have chosen 11 features to describe the anaphoric relations between an antecedent candidate a and an anaphor A. The features are presented in Table 2. Most features are domain-independent, while one, $gp_{a,A}$, is specific for the biomedical domain. Our feature set covers the basic aspects that influence anaphoric relations: the form of the anaphor's NP, string matching, semantic class matching, number agreement, and distance.

Given these features, we compute the probability P of an specific class of anaphoric relation C between a (antecedent candidate) and A (anaphor). For each pair of a given anaphor and an antecedent candidate we compute P for *C*='coreferent', *C*='biotype', and *C*='set-member'. We also compute *C*='none', that represents the probability of no relation between the NPs. P can be defined as follows:

$$P(C = \text{'class'} | f_A, f_a, hm_{a,A}, hmm_{a,A}, mm_{a,A}, num_{a,A}, sr_a, bm_{a,A}, gp_{a,A}, d_{a,A}, dm_{a,A})$$

If we were to use P as above we would suffer considerably data sparseness. In order to reduce that, we decompose the probability P and assume independence among some of the features in order to handle the sparseness of the training data. For more detail on the decomposition process refer to (Gasperin and Briscoe, 2008).

Applying Bayes' rule and selectively applying the chain rule to the above equation, as well as assuming independece among some features, we reach the following equation:

$$P(C|f_A, f_a, hm, hmm, mm, num, sr, bm, gp, d, dm) = \frac{P(C)\ P(f_A|C)\ P(f_a|C, f_A)\ P(d, dm|C, f_A, f_a)\ P(sr|C, d, dm)\ P(bm, gp|C)\ P(num|C, f_A, f_a)\ P(hm, hmm, mm|C, f_A, f_a, bm)}{P(f_A)\ P(f_a|f_A)\ P(d, dm|f_A, f_a)\ P(sr|d, dm)\ P(bm, gp)\ P(num|f_A, f_a)\ P(hm, hmm, mm|f_A, f_a, bm)} \quad (1)$$

C	"The expression of **reaper** has been shown ... **the gene** encodes ...
B	"**Drosophila gene Bok** interacts with ... expression of **Bok protein** promotes apoptosis ..."
S	"... **ced-4** and **ced-9** ... **the genes** ..."
	"... **the mammalian anti-apoptotic protein Bcl-2** ... **Bcl-2 family** ..."

Table 1: Examples of coreferent (C), associative biotype (B) and associative set-member (S) anaphoric relations

Feature	Possible values
f_A	Form of noun phrase of the anaphor A: 'pn', 'defnp', 'demnp', 'indefnp', 'quantnp', or 'np'.
f_a	Form of noun phrase of the antecedent candidate a: same values as for f_A.
$hm_{a,A}$	Head-noun matching: 'yes' if the anaphor's and the candidate's head nouns match, 'no' otherwise.
$hmm_{a,A}$	Head-modifier matching: 'yes' if the anaphor's head noun matches any of the candidate's pre-modifiers, or vice-versa, 'no' otherwise.
$mm_{a,A}$	Modifier matching: 'yes' if anaphor and candidate have at least one head modifier in common, 'no' otherwise.
$num_{a,A}$	Number agreement: 'yes' if anaphor and candidate agree in number, 'no' otherwise.
$sr_{a,A}$	Syntactic relation between anaphor and candidate: 'none', 'apposition', 'subj-obj', 'pp', and few others.
$bm_{a,A}$	Biotype matching: 'yes' if anaphor's and candidate's biotype (semantic class) match, 'no' otherwise.
$gp_{a,A}$	is biotype *gene* or *product*? 'yes' if the anaphor biotype or candidate biotype is *gene* or *product*, 'no' otherwise. This feature is mainly to distinguish which pairs can hold biotype relations.
$d_{a,A}$	Distance in sentences between the anaphor and the candidate.
$dm_{a,A}$	Distance in number of entities (markables) between the anaphor and the candidate.

Table 2: Feature set

This equation is the basis of our resolution model. We collect the statistics to train this model from a corpus annotated with anaphoric relations that we have created. The corpus is described in the next section.

3 Our corpus

There are very few biomedical corpora annotated with anaphora information, and all of them are built from paper abstracts (Cohen et al., 2005), instead of full papers. As anaphora is a phenomenon that develops through a text, we believe that short abstracts are not the best source to work with and decided to concentrate on full papers.

In order to collect the statistics to train our model, we have manually annotated anaphoric relations between biomedical entities in 5 full-text articles (approx. 33,300 words)[1], which are part of the Drosophila molecular biology literature. The corpus and annotation process are described in (Gasperin et al., 2007). To the best of our knowledge, this corpus is the first corpus of biomedical full-text articles to be annotated with anaphora information.

Before annotating anaphora, we have preprocessed the articles in order to (1) tag gene names, (2) identify all NPs, and (3) classify the NPs according to their domain type, which we call biotype. To tag all gene names in the corpus, we have applied the gene name recogniser developed by Vlachos et al. (2006). To identify all NPs, their subconstituents (head, modifiers, determiner) and broader pre- and post-modification patterns, we have used the RASP parser (Briscoe et al., 2006). To classify the NPs according to their type in biomedical terms, we have adopted the Sequence Ontology (SO)[2] (Eilbeck and Lewis, 2004). SO is a fine-grained ontology, which contains the names of practically all entities that participate in genomic sequences, besides the relations among these entities (e.g. is-a, part-of, derived-from relations). We derived from SO seven biotypes to be used to classify the entities in the text, namely: "gene", "gene product", "part of gene", "part of product", "gene variant", "gene subtype", and "gene

[1]Corpus available via the FlySlip project website http://www.wiki.cl.cam.ac.uk/rowiki/NaturalLanguage/FlySlip

[2]http://www.sequenceontology.org/

Class	Relations
coreferent	1678
biotype	274
set-member	543
discourse new	436
Total	3048
none	873,731

Table 3: Training instances, according to anaphoric class

supertype". We also created the biotype "other-bio" to be associated with noun phrases that contain a gene name (identified by the gene name recogniser) but whose head noun does not fit any of the other biotypes. All NPs were tagged with their biotypes, and NPs for which no biotypes were found were excluded.

The gene-name tags, NP boundaries and biotypes resulting from the preprocessing phase were revised and corrected by hand before the anaphoric relations were annotated.

For each biotyped NP we annotated its closest coreferent antecedent (if found) and its closest associative antecedent (if found), from one of the associative classes. From our annotation, we can infer coreference chains by merging the coreferent links between mentions of a same entity.

The annotated relations, and the features derived from them, are used as training data for the probabilistic model above. A special characteristic of data annotated with anaphora information is the overwhelming amount of negative instances, which result from the absence of an anaphoric relation between a NP that precedes an anaphoric expression and was not marked as its antecedent (nor marked as part of the same coreference chain of its antecedent). The negative instances outnumber considerably the number of positive instances (annotated cases). Table 3 presents the distribution of the cases among the classes of anaphoric relations.

To balance the ratio between positive and negative training samples, we have clustered the negative samples and kept only a portion of each cluster, proportional to its size. All negative samples that have the same values for all features are grouped together (consequently, a cluster is formed by a set of identical samples) and only one-tenth of each cluster members is kept, resulting in 85,314 negative samples. This way, small clusters (with less than 10 members), which are likely to represent noisy samples (similar to positive ones), are eliminated, and bigger clusters are shrunk; however the shape of the distribution of the negative samples is preserved. For example, our biggest cluster (feature values are: f_A='pn', f_a='pn', hm='no', hmm='no', mm='no', bm='yes', gp='yes', num='yes', sr='none', d='16<', dm='50<') with 33,998 instances is reduced to 3,399 – still considerably more numerous than any positive sample. Other works have used a different strategy to reduce the imbalance between positive and negative samples (Soon et al., 2001; Ng and Cardie, 2002; Strube et al., 2002), where only samples composed by a negative antecedent that is closer than the annotated one are considered. Our strategy is more flexible and is able to the reduce further the number of negative samples. The higher the number of negative samples, the higher the precision of the resolution, but the lower the recall.

4 Active learning

When trained using all our annotated corpus on a 10-fold cross-validation setting our anaphora resolution model, presented above, reached the results shown in Table 4[3].

We would like to improve this results without having to annotate too much more data, therefore we decided to experiment with active learning. We defined three entropy-based measures to calculate the uncertanty of our model for each decidion is makes.

[3] 'Perfect' scores shows the result of a strict evaluation, where we consider as correct all pairs that match exactly an antecedent-anaphor pair in the annotated data. On the other hand, column 'Relaxed' treats as correct also the pairs where the assigned antecedent is not the exact match in the annotated data but is coreferent with it.

Class	Perfect			Relaxed		
	P	R	F	P	R	F
coreferent	56.3	54.7	55.5	69.4	67.4	68.3
biotype	28.5	35.0	31.4	31.2	37.9	34.2
set-member	35.4	38.2	36.7	38.5	41.5	40.0
discourse new	44.3	53.4	48.4	44.3	53.4	48.4

Table 4: Performance of the probabilistic model

4.1 Uncertainty measures

In order to measure how confident our model is about the class it assigns to each candidate, and consequently the one it chooses as the antecedent of an anaphor, we experiment with the following entropy-based measures.

We first compute what we call the "local entropy" among the probabilities for each class—*P(C="coreferent")*, *P(C="biotype")*, *P(C="set-member")* and *P(C="none")*—for a given pair anaphor(A)-candidate(a), which is defined as

$$LE(A, a) = -\sum_{C} P(C) log_2 P(C) \qquad (2)$$

where $P(C)$ represents Equation 1 above, that is, the probability assigned to the anaphor-candidate relation by our probabilistic model for a particular class. The more similar the probabilities are, the more uncertain the model is about the relation, so the higher the local entropy. This measure is similar to others used in previous work for different problems.

We also compute the "global entropy" of the distribution of candidates across classes for each anaphor. The global entropy aims to combine the uncertainty information from all antecedent candidates for a given anaphor (instead of considering only a single candidate-anaphor pair as for LE). The higher the global entropy, the higher the uncertainty of the model about the antecedent for an anaphor. The global entropy combines the local entropies for all antecedent candidates of a given anaphor. We propose two versions of the global entropy measure. The first is simply a sum of the local entropies of all candidates available for a given anaphor, it is defined as

$$GE1(A) = \sum_{a} LE(A, a) \qquad (3)$$

The second version averages the local entropies across all candidates, it is defined as

$$GE2(A) = \frac{\sum_a LE(A, a)}{|a|} \qquad (4)$$

where $|a|$ corresponds to the number of candidates available for a given anaphor.

We consider that in general the further away a candidate is from the anaphor, the lower the local entropy of the pair is (given that when distance increases, the probability of the candidate not being the antecedent, *P(C="none")*, also increases), and consequently the less it contributes to the global entropy. This is the intuition behind $GE1(A)$.

However, in some cases, mainly when the anaphor is a proper name, there can be several candidates at a long distance from the anaphor that still get a reasonable probability assigned to them due to positive string matching. Therefore we decided to experiment with averaging the sum of the local probabilities by the number of candidates, so $GE2(A)$.

5 Experiments

Initially, our training data was divided in 10-folds for cross-validation evaluation of our probabilistic model for anaphora resolution. For the active learning experiments we kept the same folds, using one for the initial training, eight for the active learning phase, and the remaining one for testing. We have experimented with 10 different initial-training/active-learning/testing splits, selected randomly from all combinations of the 10 folds, and the results in this section correspond to the average of the results from the different data splits. A fold contains the positive and negative samples derived from about 270 anaphors, it contains about 7000 candidate-anaphor pairs (an average of about 26 antecedent candidates per anaphor). The anaphors that are part of each fold were randomly selected.

The purpose of our experiments is to check whether the samples selected by using the entropy-based measures described above, when added to our training data, can improve the performance of the model more than in the case of adding the same amount of randomly selected samples. For that, we computed (1) the performance of our model using one fold of training data, (2) the performance of the model over 10 iterations of active learning using each of the uncertainty measures above, and (3) the performance of the model over 10 iterations adding the same amount of randomly selected instances as for active learning. At each active learning iteration, when using $LE(A, a)$ we selected the 1500 candidate-anaphor pairs for which uncertainty was the highest, and when using $GE1(A)$ and $GE2(A)$ we selected the 50 anaphors for which the

model was most uncertain and generated the positive and negative instances that were associated to the anaphors.

We expected (2), entropy-based sample selection, to achieve better performance than (3), random sample selection, however this has not happened. The graphs in Figure 1 compare the precision, recall and F-measure scores for (2) and (3) along the 10 iterations for each class of anaphoric relation. The lines corresponding to random sampling plot the results of the experiments done in the same way as for $GE1(A)$ and $GE2(A)$, that is, where 50 anaphors are selected at each iteration, although we also tested random sampling in the $LE(A, a)$ fashion, selecting 1500 candidate-anaphor pairs.

We observe that none of the uncertainty measures that we tested have performed consistently better than random sampling. $LE(A, a)$ presents the most dramatic results, it worsens the general performance of the model for all classes, although it causes a considerable increase in precision for coreferent and set-member cases. $GE1(A)$ and $GE2(A)$ have a less clear pattern, but it is possible to notice that $GE1(A)$ tends to bring improvements in precision while $GE2(A)$ causes the opposite, improvements in recall and drops in precision.

6 Discussion

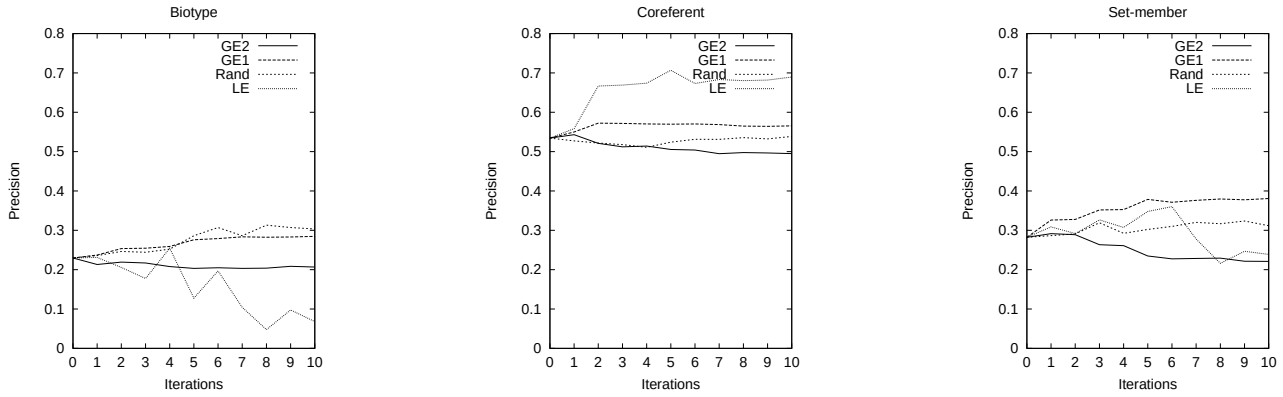
When looking at the instances selected by each active learning strategy, we observe the following. $LE(A, a)$, which considers anaphor-candidate pairs, selects mostly negative instances, given the fact that these are highly frequent. This can explain the increase in precision and drop in recall for the positive cases (observed for coreferent and set-member, the most frequent positive classes), since that is expected with the increase of negative instances.

$GE1(A)$ and $GE2(A)$ select a proportional number of positive and negative instances, since these measures consider an anaphor and all possible antecedent candidates, generating all instances that derive from each selected anaphor (usually one or two positive intances and several negative ones). However, we can observe some differences between the impact of using $GE1(A)$ and $GE2(A)$ to select instances. We observe that about 70% of the samples selected by $GE1(A)$ were proper names, while the distribution of NP types among the samples selected by $GE2(A)$ is similar to the original distribution in the data. This confirms the problem we expected to have with $GE1(A)$, since exact matches of proper names that occur at a considerable distance from the anaphor still get a higher probability assigned to them, which does not happen so often with other types of NPs. On the other hand, the correct antecedent of about 30% of $GE2(A)$-selected samples were in the same sentence as the anaphor, while the same occurs with only 8% of $GE1(A)$-selected samples. $GE2(A)$ behaviour in this case is counter intuitive, since antecedents in the same sentence should be found by the model with lower uncertainty than antecedents further away from the anaphor. Another counter intuitive behaviour of $GE2(A)$ is that only 3% of the selected anaphors have no string matching with their antecedents (33% have no head-noun matching), while these cases correspond to about 30% of samples selected by $GE1(A)$ (62% of samples have no head-noun matching). We expected samples involving no string matching to be selected because they are usually the ones the model is mostly uncertain about.

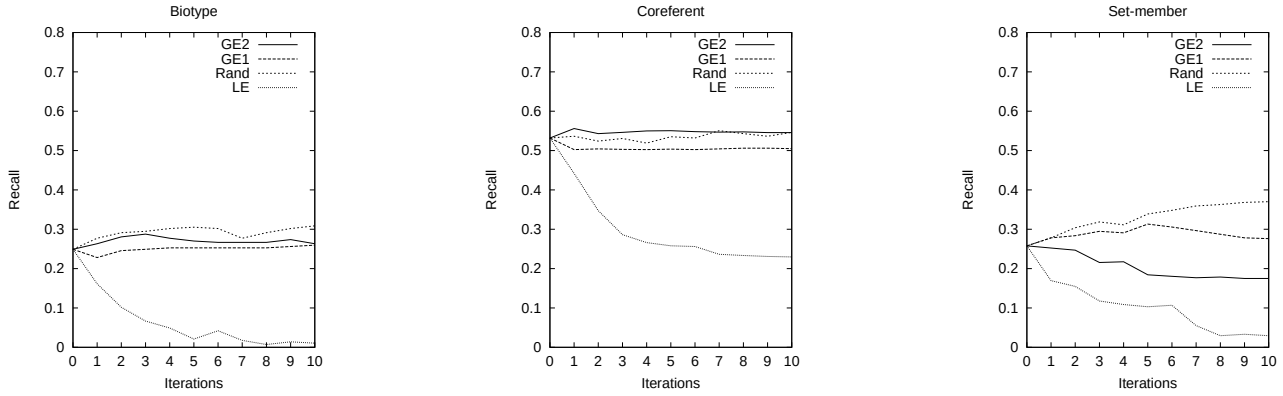
Despite the different behaviour among the measures none was successful in improving the performance of the model in relation to the performance of random sampling.

While entropy-based measures for sample selection seem the obvious option given that we use a probabilistic model, they did not give positive results in our case. A future study of different ways to combine the local entropies is necessary, as well as the study of other non-entropy-based measures for sample selection.

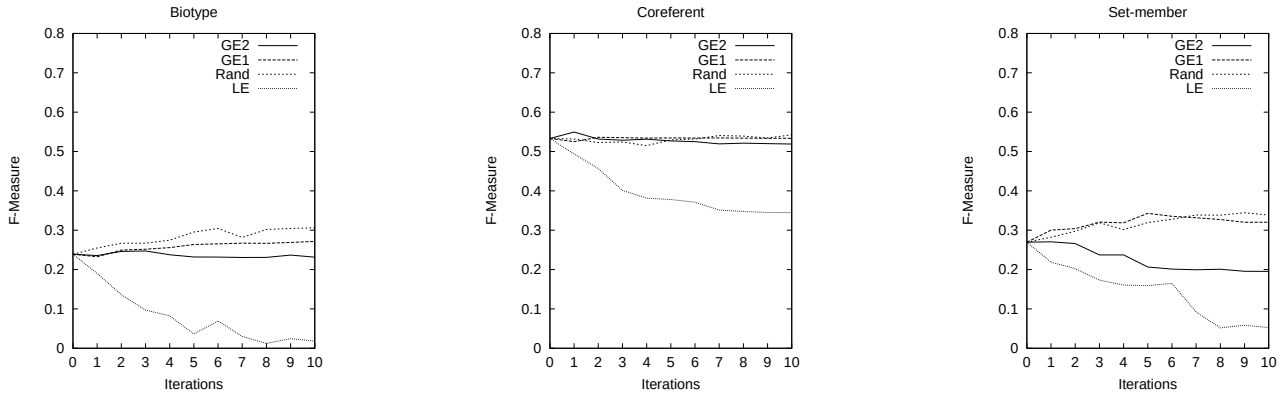
The main difference between our application of active learning to anaphora resolution and previous successful applications of active learning to other tasks is the amount of probabilities involved in the calculation of the uncertainty of the model. We believe this is the reason why our active learning experiments were not succesfull. While, for example, name entity recognition involves a binary decision, and parse selection involves a few parsing options, in our case there are several antecedent candidates to be considered. For anaphora resolution, when using a pairwise resolution model, it is necessary to combine the predictions for one candidate-anaphor



(a) Precision



(b) Recall



(c) F-measure

Figure 1: Graphs of the performance of active learning using $LE(A, a)$, $GE1(A)$, $GE2(A)$ and random sampling.

pair to the others in order to predict the global uncertainty of the model.

Acknowledgments

This work was part of the BBSRC-funded FlySlip project. Caroline Gasperin was funded by a CAPES award from the Brazilian government. Thanks to Ted Briscoe for comments on this manuscript.

References

Jason Baldridge and Miles Osborne. 2003. Active learning for hpsg parse selection. In Walter Daelemans and Miles Osborne, editors, *Proceedings of CoNLL-2003*, pages 17–24. Edmonton, Canada.

Edward J. Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the RASP system. In *Proceedings of ACL-COLING 06*, Sydney, Australia.

K. Bretonnel Cohen, Lynne Fox, Philip Ogren, and Lawrence Hunter. 2005. Corpus design for biomedical natural language processsing. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases*, Detroit.

Karen Eilbeck and Suzanna E. Lewis. 2004. Sequence ontology annotation guide. *Comparative and Functional Genomics*, 5:642–647.

Caroline Gasperin and Ted Briscoe. 2008. Statistical anaphora resolution in biomedical texts. In *Proceedings of COLING 2008*, Manchester, UK.

Caroline Gasperin, Nikiforos Karamanis, and Ruth Seal. 2007. Annotation of anaphoric relations in biomedical full-text articles using a domain-relevant scheme. In *Proceedings of DAARC 2007*, pages 19–24, Lagos, Portugal.

Niyu Ge, John Hale, and Eugene Charniak. 1998. A statistical approach to anaphora resolution. In *Proceedings of the Sixth Workshop on Very Large Corpora - COLING-ACL'98*, Montreal, Canada.

Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of ACL 2002*, Philadelphia.

Massimo Poesio and Renata Vieira. 1998. A corpus-based investigation of definite description use. *Computational Linguistics*, 24(2):183–216.

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 ACL 2004*, Barcelona.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Michael Strube, Stefan Rapp, and Christoph Mller. 2002. The influence of minimum edit distance on reference resolution. In *Proceedings of the EMNLP 2002*, Philadelphia.

Min Tang, Xiaoqiang Luo, and Salim Roukos. 2002. Active learning for statistical natural language parsing. In *Proceedings of ACL 2002*, pages 120–127, Philadelphia, Pennsylvania. Association for Computational Linguistics.

Andreas Vlachos and Caroline Gasperin. 2006. Bootstrapping and evaluating named entity recognition in the biomedical domain. In *Proceedings of BioNLP at HLT-NAACL 2006*, pages 138–145, New York.

On Proper Unit Selection in Active Learning: Co-Selection Effects for Named Entity Recognition

Katrin Tomanek[1*] **Florian Laws**[2*] **Udo Hahn**[1] **Hinrich Schütze**[2]

[1]Jena University Language & Information Engineering (JULIE) Lab
Friedrich-Schiller-Universität Jena, Germany
{katrin.tomanek|udo.hahn}@uni-jena.de

[2]Institute for Natural Language Processing, Universität Stuttgart, Germany
{fl|hs999}@ifnlp.org

Abstract

Active learning is an effective method for creating training sets cheaply, but it is a biased sampling process and fails to explore large regions of the instance space in many applications. This can result in a missed cluster effect, which signficantly lowers recall and slows down learning for infrequent classes. We show that missed clusters can be avoided in sequence classification tasks by using sentences as natural multi-instance units for labeling. Co-selection of other tokens within sentences provides an implicit exploratory component since we found for the task of named entity recognition on two corpora that entity classes co-occur with sufficient frequency within sentences.

1 Introduction

Active learning (AL) has been shown to be an effective approach to reduce the amount of data needed to train an accurate statistical classifier. AL selects highly informative examples from a pool of unlabeled data and prompts a human annotator for the labels of these examples. The newly labeled examples are added to a training set used to build a statistical classifier. This classifier is in turn used to assess the informativeness of further examples. Thus, a select-label-retrain loop is formed that quickly selects hard to classify examples, honing in on the decision boundary (Cohn et al., 1996).

A fundamental characteristic of AL is the fact that it constitutes a biased sampling process. This is so by design, but the bias can have an undesirable consequence: partial coverage of the instance space. As a result, classes or clusters within classes may be completely missed, resulting in low recall or slow learning progress. This has been called the *missed cluster effect* (Schütze et al., 2006). While AL has been studied for a range of NLP tasks, the missed cluster problem has hardly been addressed.

This paper studies the *missed class effect*, a special case of the missed cluster effect where complete classes are overlooked by an active learner. The missed class effect is the result of insufficient exploration before or during a mainly exploitative AL process. In AL approaches where exploration is only addressed by an initial seed set, poor seed set construction gives rise to the missed class effect.

We focus on the missed class effect in the context of a common NLP task: named entity recognition (NER). We show that for this task the missed class effect is avoided by increasing the sampling granularity from single-instance units (i.e., tokens) to multi-instance units (i.e., sentences). For AL approaches to NER, sentence selection recovers better from unfavorable seed sets than token selection due to what we call the *co-selection effect*. Under this effect, a non-targeted entity class co-occurs in sentences that were originally selected because of uncertainty on tokens of a different entity class.

The rest of the paper is structured as follows: Section 2 introduces the missed class effect in detail. Experiments which demonstrate the co-selection effect achieved by sentence selection for NER are described in Section 3 and their results presented in Section 4. We draw conclusions in Section 5.

[*] Both authors contributed equally to this work.

Proceedings of the NAACL HLT Workshop on Active Learning for Natural Language Processing, pages 9–17,
Boulder, Colorado, June 2009.

2 The Missed Class Effect

This section first describes the missed class effect. Then, we discuss several factors influencing this effect, focusing on co-selection, a natural phenomenon in common NLP applications of AL.

2.1 Sampling bias and misguided AL

The distribution of the labeled data points obtained with an active learner deviates from the true data distribution. While this sampling bias is intended and accounts for the effectiveness of AL, it also poses challenges as it leads to classifiers that perform poorly in some regions, or *clusters*, of the example space. In the literature, this phenomenon has been described as the *missed cluster effect* (Schütze et al., 2006; Dasgupta and Hsu, 2008)

In this context, we must distinguish between exploration and exploitation. By design, AL is a highly exploitative strategy: regions around decision boundaries are inspected thoroughly so that decision boundaries are learned well, but regions far from any of the initial decision boundaries remain unexplored.

An exploitative sampling approach thus has to be combined with some kind of exploratory strategy to make sure the example space is adequately covered. A common approach is to start an AL process with an initial seed set that accounts for the exploration step. However, a seed set which is not representative of the example space may completely misguide AL — at least when no other explorative techniques are applied as a remedy. While approaches to balancing exploration and exploitation (Baram et al., 2003; Dasgupta and Hsu, 2008; Cebron and Berthold, 2009) have been discussed, we here focus on a "pure" AL scenario where exploration takes only place in the beginning by a seed set. In summary, the missed clusters are the result of a scenario where poor exploration is combined with exclusively exploitative sampling.

Why is AL an exploitative sampling strategy? AL selects data points based on the confidence of the active learner. Assume an initial seed set that does not contain examples of a specific cluster. This leads to an initial active learner that is mistakenly overconfident about the class membership of instances in this missed cluster. Far away from the decision boundary, the active learner assumes a high confidence for all instances in that cluster, even if they are in fact misclassified. Consequently, the active learner will fail to select these instances for long until some redirection impulse is received (if at all).

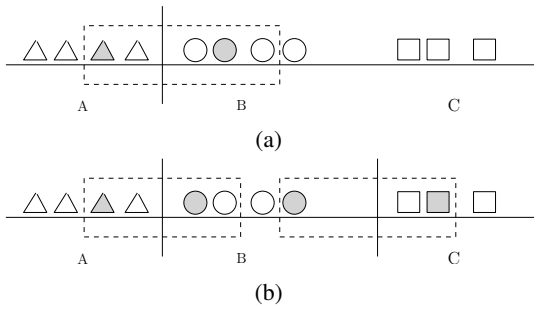


Figure 1: Illustration of the missed cluster effect in a 1-d scenario. Shaded points are contained in the seed set, vertical lines are final decision boundaries, and dashed rectangles mark the explored regions

To give an example, let us consider a simple 1-d toy scenario with examples from three clusters A, B, and C as shown in Figure 1. In scenario *(a)*, AL is started from a seed set including one example of clusters A and B only. In subsequent rounds, AL will select examples in these clusters only (shown as the dashed box in the figure). Examples in cluster C are ignored as they are far from the initial decision boundary. Eventually, a decision boundary is fixed as shown by the vertical line which indicates that this AL process has completely overlooked examples from cluster C.

Assuming that the examples fall in two classes $X_1 = \{A \cup C\}$ and $X_2 = \{B\}$ the learned classifier has low recall for class X_1 and relatively low precision for class X_2 as it erroneously assigns examples of cluster C to class X_2. In a related scenario with three classes $X_1 = \{A\}$, $X_2 = \{B\}$, and $X_3 = \{C\}$ this would even mean that the classifier is not at all aware about the third class resulting in the *missed class* problem.

A more representative seed set circumvents this problem. Given a seed set including one example of each cluster, AL might find a second decision boundary[1] between clusters B and C because it is now aware of examples from C. Figure 1(b) shows a possible result of AL on this seed set.

The missed cluster effect can be understood as the generalized problem. A special case of it is the

[1] Assuming a classifier that can learn several boundaries.

missed class effect as shown in the previous example. In general, it has the same causes (insufficient exploration and misguided exploitation), but is easier to test. Often we know (at least the number of) all classes under scrutiny, while we usually cannot assume all clusters in the feature space to be known. In this paper, we focus on the missed class effect, i.e., scenarios where classes are overlooked by a misguided AL process resulting in a slow (active) learning progress.

2.2 Factors influencing the missed class effect

AL in a practical scenario is subject to several factors which mitigate or intensify the missed class effect described before. In the following, we describe three such factors, with a special focus on the co-selection effect, which we claim to significantly mitigate the missed class effect in a specific type of NLP tasks, sequence learning problems such as NER or POS tagging.

Class imbalance Many studies on AL for NLP tasks assume that AL is started from a randomly drawn seed set. Such a seed set can be problematic when the class distribution in the data is highly skewed. In this case, "rare" classes might not be represented in the seed set, increasing the chance to completely miss out such a class using AL. When classes are relatively frequent, an active learner — even when started from an unfavorable seed set — might still mistake an example of one class for an uncertain example of a different class and consequently select it. Thereby, it can acquire information about the former class "by accident" leading to sudden and rapid discovery of the newly-found class. However, in the case of extreme class imbalance this is very unlikely. Severe class imbalance intensifies the missed cluster effect.

Similarity of considered classes If, e.g., two of the classes to be learned, say X_i and X_j, are harder to discriminate than others, or if the data contains lots of noise, an active learner is more likely to select some instances of X_i if at least its "similar" counterpart X_j was represented in the seed set. Hence, it may mistake the instances of X_i and X_j before it has acquired enough information to discriminate between them. So, under certain situations similarity of classes can mitigate the missed class effect.

The co-selection effect Many NLP tasks are sequence learning problems including, e.g., POS tagging, and named entity recognition. Sequences are consecutive text tokens constituting linguistically plausible chunks, e.g., sentences. Algorithms for sequence learning obviously work on sequence data, so respective AL approaches need to select complete sequences instead of single text tokens (Settles and Craven, 2008). Furthermore, sentence selection has been preferred over token selection in other works with the argument that the manual annotation of single, possibly isolated tokens is almost impossible or at least extremely time-consuming (Ringger et al., 2007; Tomanek et al., 2007).

Within such sequences, instances of different classes often co-occur. Thus, an active learner that selects uncertain examples of one class gets examples of a second class as an unintended, yet positive side effect. We call this the *co-selection effect*. As a result, AL for sequence labeling is not "pure" exploitative AL, but implicitly comprises an exploratory aspect which can substantially reduce the missed class problem. In scenarios where we cannot hope for such a co-selection, we are much more likely to have decreased AL performance due to missed clusters or classes.

3 Experiments

We ran several experiments to investigate how the sampling granularity, i.e. the size of the selection unit, influences the missed class effect. AL based on token selection (T-AL) is compared to AL based on sentence selection (S-AL). Although our experiments are certainly also subject to the other factors mitigating the missed class effect (e.g. similarity of classes), the main focus of the experiments is on the co-selection effect that we expected to observe in S-AL. Several scenarios of initial exploration were simulated by seed sets of different characteristics. The experiments were run on synthetic and real data in the context of named entity recognition (NER).

3.1 Classifiers and active learning setup

The active learning approach used for both S-AL and T-AL is based on uncertainty sampling (Lewis and Gale, 1994) with the *margin* metric (Schein and Ungar, 2007) as uncertainty measure. Let c and c'

be the two most likely classes predicted for token x_j with $\hat{p}_{c,x_j}$ and $\hat{p}_{c',x_j}$ being the associated class probabilities. The per-token margin is calculated as $M = |\hat{p}_{c,x_j} - \hat{p}_{c',x_j}|$.

For T-AL, the sampling granularity is the token, while in S-AL, complete sentences are selected. For S-AL, the margins of all tokens in a sentence are averaged and the aggregate margin is used to select sentences. We chose this uncertainty measure for S-AL for better comparison with T-AL. In either case, examples (tokens or sentences) with a small margin are preferred for selection. In every iteration, a batch of examples is selected: 20 sentences for S-AL, 200 tokens for T-AL.

Bayesian logistic regression as implemented in the BBR classification package (Genkin et al., 2007) with out-of-the-box parameter settings was used as base learner for T-AL. For S-AL, a linear-chain Conditional Random Field (Lafferty et al., 2001) is employed as implemented in MALLET (McCallum, 2002). Both base learners employ standard features for NER including the lexical token itself, various orthographic features such as capitalization, the occurrence of special characters like hyphens, and context information in terms of features of neighboring tokens to the left and right of the current token.

3.2 Data sets

We used three data sets in our experiments. Two of them (ACE and PBIO) are standard data sets. The third (SYN) is a synthetic set constructed to have specific characteristics. For simplicity, we consider only scenarios with two entity classes, a majority class (MAJ) and a minority class (MIN). We discarded all other entity annotations originally contained in the corpus assigning the OUTSIDE class.[2]

The first data set (PBIO) is based on the annotations of the PENNBIOIE corpus for biomedical entity extraction (Kulick et al., 2004). As PENNBIOIE makes fine-grained and subtle distinctions between various subtypes of classes irrelevant for this study, we combined several of the original classes into two entity classes: The majority class consists of the three original classes 'gene-protein', 'gene-generic', and 'gene-rna'. The minority class consists of the original and similar classes 'variation-type' and 'variation-event'. All other entity labels were replaced by the OUTSIDE class.

The second data set (ACE) is based on the newswire section of the ACE 2005 Multilingual Training Corpus (Walker et al., 2006). We chose the 'person' class as majority class and the 'organization' class as the minority class. Again, all other classes are mapped to OUTSIDE.

The synthetic data set (SYN) was constructed by combining the sentences from the original ACE and PENNBIOIE corpora. The 'person' class constitutes the minority class, the very similar classes 'malignancy' and 'malignancy-type' were merged to form the majority class. All other class labels were set to OUTSIDE. SYN's construction was motivated by the following characteristics of the new data set which would make the appearance of the missed class effect very likely for insufficient exploration scenarios:
(i) absence of inner-sentence entity class correlation to ensure that sentences contain either mentions of only a single entity class or no mentions at all.
(ii) marked entity class imbalance between the majority and minority classes
(iii) dissimilar surface patterns of entity mentions of the two entity classes with the rationale that class similarity will be low.

Table 1 summarizes characteristics of the data sets. While SYN exhibits high imbalance (e.g., 1:9.4 on the token level), PBIO and ACE are moderately skewed. In PBIO, the number of sentences containing any entity mention is relatively high compared to ACE or SYN. For our experiments, the corpora were randomly split in a pool for AL and a test set for performance evaluation.

Inner-sentence entity class co-occurrence We have described co-selection as a potential mitigating factor for the missed class effect in Section 2. For this effect to occur, there must be some correlation between the occurrence of entity mentions of the MAJ class with those from MIN.

Table 2 shows correlation statistics based on the χ^2 measure. We found strong correlation in all three corpora[3]: For ACE and PBIO, the correlation is positive; for SYN it is negative so when a sentence in SYN contains a majority class entity mention, it is

[2]The OUTSIDE class marks that a token is not part of an named entity.

[3]All correlations are statistically significant ($p < 0.01$).

	PBIO	ACE	SYN
sentences (all)	11,164	2,642	13,804
sentences (MAJ)	7,075	767	5,667
sentences (MIN)	2,156	974	974
MIN-MAJ ratio	1 : 3.3	1 : 1.3	1 : 5.8
tokens (all)	277,053	66,752	343,773
tokens (MAJ)	17,928	2,008	18,959
tokens (MIN)	4,079	1,822	2,008
MIN-MAJ ratio	1 : 4.4	1 : 1.1	1 : 9.4

Table 1: Characteristics of the data sets; "sentences (MAJ)", e.g., specifies the number of sentences containing mentions of the majority class.

	PBIO	ACE	SYN
χ^2	132.34	6.07	727
$P(MIN\|MAJ)$	0.26	0.31	0.0

Table 2: Co-occurrence of entity classes in sentences

highly unlikely that it also contains a minority entity. In fact, it is impossible by construction of the data set. Further, this table shows the probability that a sentence containing the majority class also contains the minority class. As expected, this is exactly 0 for SYN, but significantly above 0 for PBIO and ACE.

3.3 Seed sets

Selection of an appropriate seed set for the start of an AL process is important to the success of AL. This is especially relevant in the case of imbalanced classes because a typically small random sample will possibly not contain any example of the rare class. We constructed different types of seed sets (whose naming intentionally reflects the use of the entity classes from Section 3.2) to simulate different scenarios of ill-managed initial exploration. All seed sets have a size of 20 sentences. The *RANDOM* set was randomly sampled, the *MAJ* set is made of sentences containing at least one majority class entity, but no minority class entity. Accordingly, *MIN* is densely populated with minority entities. Finally, *OUTSIDE* contains only sentences without entity mentions.

One could think of the OUTSIDE and MAJ seed sets of cases where a random seed set selection has unluckily produced an especially bad seed set. MIN serves to demonstrate the opposite case. For each type of seed set, we sampled ten independent versions to calculate averages over several AL runs.

3.4 Cost measure

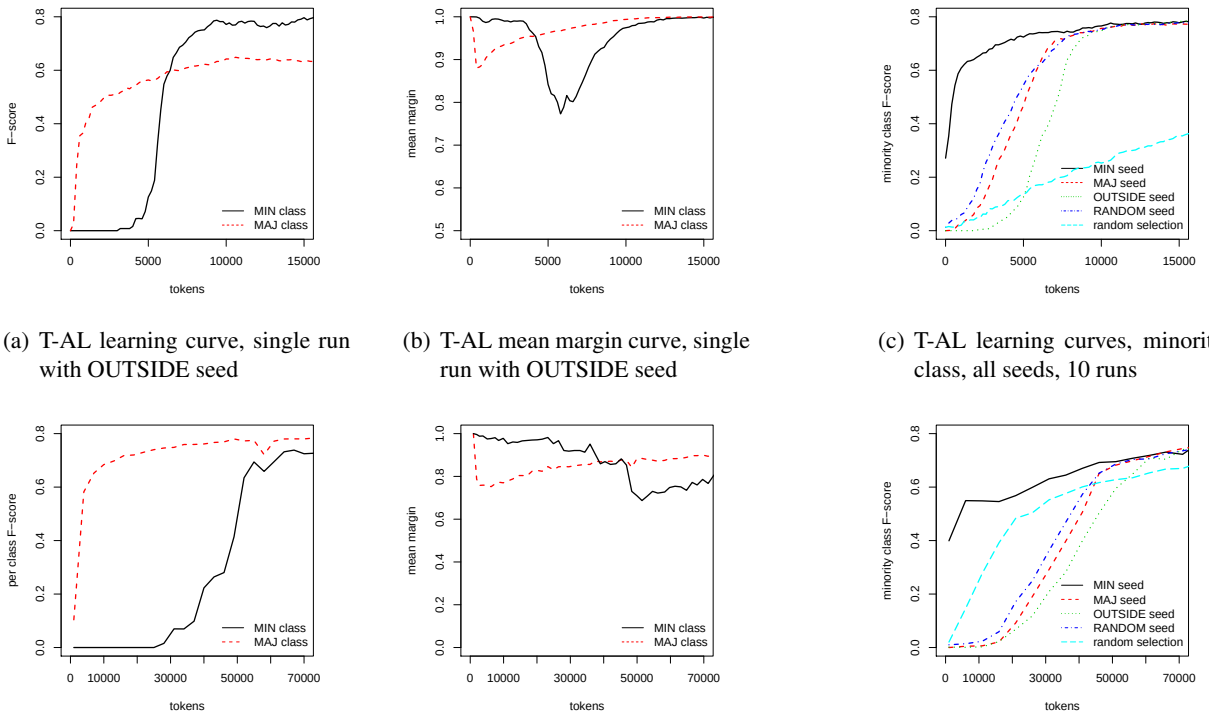
The success of AL is usually measured as reduction of annotation effort according to some cost measure. Traditionally, the most common cost measure considers a unit cost per annotated token, which favors AL systems that select individual tokens. In a real annotation setting, however, it is unnatural, and therefore hard for humans to annotate single, possibly isolated tokens, leading to bad annotation quality (Hachey et al., 2005; Ringger et al., 2007). When providing context, the question arises whether the annotator can label several tokens present in the context (e.g., an entire multi-token entity or even the whole sentence) at little more cost than annotating a single token. Thus, assigning a linear cost of n to a sentence where n is the sentence's length in tokens seems to unfairly disadvantage sentence-selection AL setups.

However, more work is needed to find a more realistic cost measure. At present there is no other generally accepted cost measure than unit cost per token, so we report costs using the token measure.

4 Results

This section presents the results of our experiments on the missed class effect in two different AL scenarios, i.e., sentence selection (S-AL) and token selection (T-AL). The AL runs were stopped when convergence on the minority class F-score was achieved. This was done because early AL iterations before the convergence point are most important and representative for a real-life scenario where the pool is extremely large, so that absolute convergence of the classifier's performance will never be reached.

The learning curves in Figures 2, 3, and 4 reveal general characteristics of S-AL compared to T-AL. For S-AL, the number of tokens on the x-axis is the total number of tokens in the sentences labeled so far. While S-AL generally yields higher F-scores, T-AL converges much earlier when counted in terms of tokens. The reason for this is that T-AL can select uncertain data more specifically. In contrast, S-AL also selects tokens that the classifier can already classify reliably – these tokens are selected because they co-occur in a sentence that also contains an uncertain token. Whether T-AL is really more efficient clearly depends on the cost-metric applied (cf. Sec-



(a) T-AL learning curve, single run with OUTSIDE seed

(b) T-AL mean margin curve, single run with OUTSIDE seed

(c) T-AL learning curves, minority class, all seeds, 10 runs

(d) S-AL learning curve, single run with OUTSIDE seed

(e) S-AL mean margin curve, single run with OUTSIDE seed

(f) S-AL learning curves, minority class, all seeds, 10 runs

Figure 2: Results on SYN corpus for token selection (a,b,c) and sentence selection (d,e,f)

tion 3.4). Since the focus of this paper is on comparing the missed class effect in a sentence and a token selection AL setting (T-AL and S-AL) we apply the straight-forward token measure.

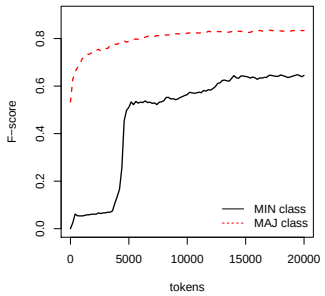
4.1 The pathological case

Figure 2 shows results on the SYN corpus for T-AL (upper row) and S-AL (lower row). Figures 2(a) and 2(d) show the minority and majority class learning curves for a single run starting from the OUTSIDE seed set, which was particularly problematic on SYN. (We show single runs to give a better picture of what happens during the selection process.) The figures show that for both AL scenarios, the OUTSIDE seed set caused the active learner to focus exclusively on the majority class and to completely ignore the minority class for many AL iterations (almost 30,000 tokens for S-AL and over 4,000 tokens for T-AL). Had we stopped the AL process before this turning point, the classifier's performance on the majority entity class would have been reasonably high while the minority class would not have been learned at all — which is precisely the definition of an (initially) missed class.

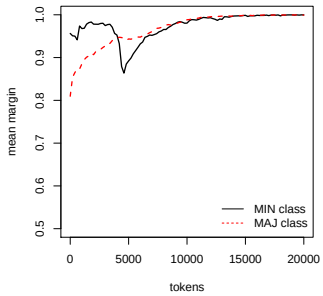
Figures 2(b) and 2(e) show the corresponding mean margin plots of these AL runs, indicating the confidence of the classifier on each class. The mean margin is calculated as the average margin over tokens in the remaining pool, separately for each true class label.[4] As expected, the active learner is overconfident but wrong on instances of the minority class (assigning them to the OUTSIDE class, we assume). Only after some time, margin scores on minority class tokens start decreasing. This happens because from time to time minority class examples are mistakenly considered as majority class examples with low confidence and thus selected by accident. Lowered minority class confidence then causes the selection of further minority class examples, resulting in a turning point with a steep slope of the minority class learning curve.

Consequences of seed set selection We compare the minority class learning curves for all types of

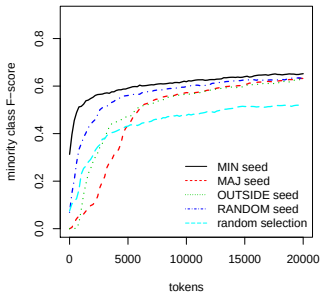
[4]Note that in a real, non-simulation active learning task, the true class labels would be unknown.



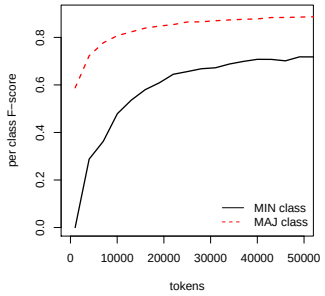
(a) T-AL learning curve, single run with MAJ seed



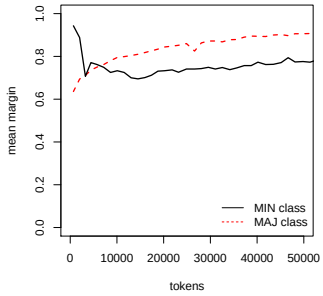
(b) T-AL mean margin curve, single run with MAJ seed



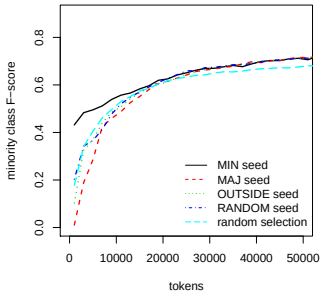
(c) T-AL learning curves, minority class, all seeds, 10 runs



(d) S-AL learning curve, single run with MAJ seed



(e) S-AL mean margin curve, single run with MAJ seed



(f) S-AL learning curves, minority class, all seeds, 10 runs

Figure 3: Results on PBIO corpus for token selection (a,b,c) and sentence selection (d,e,f)

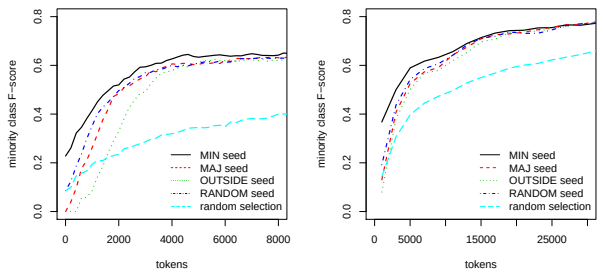
seed sets and for random selection (cf. Figures 2(c) and 2(f)), now averaged over 10 runs. On S-AL all but the MIN seed set were inferior to random selection. Even the commonly used random seed set selection is problematic because the minority class is so rare that there are random seed sets without any example of the minority class.

On T-AL, all seed sets are better than random selection. This, however, is because random selection is an extremely weak baseline for T-AL due to the token distribution (cf. Table 1). Still, the RANDOM, MAJ, and OUTSIDE seed sets are significantly worse than a seed set which covers the minority class well. Note that the majority class learning curves are relatively invariant against different seed sets. The minority class seed set does have some negative impact on initial learning progress on the majority class (not shown here), but the impact is rather small. Because of the higher frequency of the majority class, the classifier soon finds majority class examples to compensate for the seed set by chance or class similarity.

4.2 Missed class effect mitigated by co-selection

Results on PBIO corpus On the PBIO corpus, where minority and majority class entity mentions naturally co-occur on the sentence level, we get a different picture. Figure 3 shows the learning (3(a), 3(d)) and mean margin (3(b), 3(e)) curves for the MAJ seed set. T-AL still exhibits the missed class effect on this seed set. The minority class learning curve again has a delayed slope and high mean margin scores of minority tokens at the beginning, resulting in insufficient selection and slow learning. S-AL, on the other hand, does not really suffer from the missed class effect: minority class entity mentions are co-selected in sentences which were chosen due to uncertainty on majority class tokens. Minority class mean margin scores quickly fall, reinforcing selection for minority class entities. Learning curves for minority and majority classes run approximately in parallel.

Figure 3(f) shows that all seed sets perform quite similar for S-AL. MIN unsurprisingly is a bit better. With the other seed sets, S-AL performance is com-



(a) T-AL (b) S-AL

Figure 4: Minority class learning curves for all seeds on ACE averaged over 10 runs

parable to random selection. On the PBIO corpus, random selection is a strong baseline as almost every sentence contains an entity mention — which is not the case for SYN and ACE (cf. Table 1). As there is no co-selection effect for T-AL, the MAJ and OUTSIDE seed sets also here are subject to the missed class problem (Figure 3(c)), although not as severely as on the SYN corpus.

Results on ACE corpus Figure 4 shows learning curves averaged over 10 runs on ACE. Overall, the missed class effect is less pronounced on ACE compared to PBIO. Still, co-selection avoids a good portion of the missed class effect on S-AL — all seed sets yield results much better than random selection right from the beginning.

On T-AL, the OUTSIDE seed set has a marked negative effect. However, while different seed sets still have visible differences in learning performance, the magnitude of the effect is smaller than on PBIO. It is difficult to find the exact reasons in a non-synthetic, natural language corpus where a lot of different effects are intermingled. One might assume higher class similarity between the majority ("persons") and the minority ("organizations") classes on the ACE corpus than, e.g., on the PBIO corpus. Moreover, there is hardly any imbalance in frequency between the two entity classes on the ACE corpus. We briefly discussed such influencing factors possibly mitigating the missed class effect in Section 2.2.

4.3 Discussion

To summarize, on a synthetic corpus (SYN) the missed class effect can be well studied in both AL scenarios, i.e., S-AL and T-AL. Moving from a relatively controlled, synthetic corpus (extreme class imbalance, no inner-sentence co-occurrence between entity classes, quite different entity classes) to more realistic corpora, effects generally mix a bit due to different degrees of class imbalance and probably higher similarity between entity classes.

Our experiments unveil that co-selection in S-AL effectively helps avoid dysfunctional classifiers that insufficiently explore the instance space due to a disadvantageous seed set. In contrast, AL based on token-selection (T-AL) cannot recover from insufficient exploration as easy as AL with sentence-selection and is thus more sensitive to the missed class effect.

5 Conclusion

We have shown that insufficient exploration in the initial stages of active learning gives rise to regions of the sample space that contain missed classes that are incorrectly classified. This results in low classification performance and slow learning progress. Comparing two sampling granularities, tokens *vs.* sentences, we found that the missed class effect is more severe when isolated tokens instead of sentences are selected for labeling.

The missed class problem in sequence classification tasks can be avoided using sentences as natural multi-instance units for selection and labeling. Using multi-instance units, co-selection of other tokens within sentences provides an implicit exploratory component. This solution is effective if classes co-occur sufficiently within sentences which is the case for many real-life entity recognition tasks.

While other work has proposed sentence selection in AL for sequence labeling as a means to ease and speed up annotation, we have gathered here additional motivation from the perspective of robustness of learning. Future work will compare the beneficial effect introduced by co-selection with other forms of exploration-enabled active learning.

Acknowledgements

The first and the third author were funded by the German Ministry of Education and Research within the StemNet project (01DS001A-C) and by the EC within the BOOTStrep project (FP6-028099).

References

Yoram Baram, Ran El-Yaniv, and Kobi Luz. 2003. Online choice of active learning algorithms. In *ICML '03: Proceedings of the 20th International Conference on Machine Learning*, pages 19–26.

Nicolas Cebron and Michael R. Berthold. 2009. Active learning for object classification: From exploration to exploitation. *Data Mining and Knowledge Discovery*, 18(2):283–299.

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4:129–145.

Sanjoy Dasgupta and Daniel Hsu. 2008. Hierarchical sampling for active learning. In *ICML '08: Proceedings of the 25th International Conference on Machine Learning*, pages 208–215.

Alexander Genkin, David D. Lewis, and David Madigan. 2007. Large-scale Bayesian logistic regression for text categorization. *Technometrics*, 49(3):291–304.

Ben Hachey, Beatrice Alex, and Markus Becker. 2005. Investigating the effects of selective sampling on the annotation task. In *CoNLL '05: Proceedings of the 9th Conference on Computational Natural Language Learning*, pages 144–151.

Seth Kulick, Ann Bies, Mark Liberman, Mark Mandel, Ryan T. McDonald, Martha S. Palmer, and Andrew Ian Schein. 2004. Integrated annotation for biomedical information extraction. In *Proceedings of the HLT-NAACL 2004 Workshop 'Linking Biological Literature, Ontologies and Databases: Tools for Users'*, pages 61–68.

John D. Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01: Proceedings of the 18th International Conference on Machine Learning*, pages 282–289.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12.

Andrew McCallum. 2002. MALLET: A machine learning for language toolkit. `http://mallet.cs.umass.edu`.

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 at ACL-2007*, pages 101–108.

Andrew Schein and Lyle Ungar. 2007. Active learning for logistic regression: An evaluation. *Machine Learning*, 68(3):235–265.

Hinrich Schütze, Emre Velipasaoglu, and Jan Pedersen. 2006. Performance thresholding in practical text classification. In *CIKM '06: Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, pages 662–671.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *EMNLP '08: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079.

Katrin Tomanek, Joachim Wermter, and Udo Hahn. 2007. An approach to text corpus construction which cuts annotation costs and maintains reusability of annotated data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 486–495.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. *ACE 2005 Multilingual Training Corpus*. Linguistic Data Consortium, Philadelphia.

Estimating Annotation Cost for Active Learning in a Multi-Annotator Environment

Shilpa Arora, Eric Nyberg and Carolyn P. Rosé
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{shilpaa,ehn,cprose}@cs.cmu.edu

Abstract

We present an empirical investigation of the annotation cost estimation task for active learning in a multi-annotator environment. We present our analysis from two perspectives: selecting examples to be presented to the user for annotation; and evaluating selective sampling strategies when actual annotation cost is not available. We present our results on a movie review classification task with rationale annotations. We demonstrate that a combination of instance, annotator and annotation task characteristics are important for developing an accurate estimator, and argue that both correlation coefficient and root mean square error should be used for evaluating annotation cost estimators.

1 Introduction

Active Learning is the process of selectively querying the user to annotate examples with the goal of minimizing the total annotation cost. Annotation cost has been traditionally measured in terms of the number of examples annotated, but it has been widely acknowledged that different examples may require different annotation effort (Settles et al., 2008; Ringger et al., 2008).

Ideally, we would use actual human annotation cost for evaluating selective sampling strategies, but this will require conducting several user studies, one per strategy on the same dataset. Alternatively, we may be able to simulate the real user by an annotation cost estimator that can then be used to evaluate several selective sampling strategies without having to run a new user study each time. An annotation cost estimator models the characteristics that can differentiate the examples in terms of their annotation time. The characteristics that strongly correlate with the annotation time can be used as a criterion in selective sampling strategies to minimize the total annotation cost.

In some domains, the annotation cost of an example is known or can be calculated exactly before querying the user. For example, in biological experiments it might be calculable from the cost of the equipment and the material used (King et al., 2004). In NLP, sometimes a simplifying assumption is made that the annotation cost for an example can be measured in terms of its length (e.g. seconds of voicemail annotated (Kapoor et al., 2007); number of tokens annotated (Tomanek et al., 2007)). Another assumption is that the number of user annotation actions can be used as a proxy for annotation cost of an example (e.g. number of brackets added for parsing a sentence (Hwa, 2000); number of clicks for correcting named entities (Kristjannson et al., 2004)). While these are important factors in determining the annotation cost, none of them alone can fully substitute for the actual annotation cost. For example, a short sentence with a lot of embedded clauses may be more costly to annotate than a longer sentence with simpler grammatical structure. Similarly, a short sentence with multiple verbs and discontinuous arguments may take more time to annotate with semantic roles than a longer sentence with a single verb and simple subject-verb-object structure (Carreras and Márquez, 2004).

What further complicates the estimation of annotation cost is that even for the same example, annotation cost may vary across annotators (Settles et al., 2008). For example, non-native speakers of English were found to take longer time to annotate part of

Proceedings of the NAACL HLT Workshop on Active Learning for Natural Language Processing, pages 18–26,
Boulder, Colorado, June 2009.

speech tags (Ringger et al., 2008). Often multiple annotators are used for creating an annotated corpus to avoid annotator bias, and we may not know all our annotators beforehand. Annotation cost also depends on the user interface used for annotation (Gweon et al., 2005), and the user interface may change during an annotation task. Thus, we need a general annotation cost estimator that can predict annotation cost for a given annotator and user interface. A general estimator can be built by using annotator and user interface characteristics in addition to the instance characteristics for learning an annotation cost model, and training on data from multiple annotators and multiple user interfaces. Such a general estimator is important for active learning research where the goal is to compare selective sampling strategies independent of the annotator and the user interface.

In this work, we investigate the annotation cost estimation problem for a movie review classification task in a multi-annotator environment with a fixed user interface. We demonstrate that a combination of instance, annotation task and annotator characteristics is important for accurately estimating the annotation cost. In the remainder of the paper, we first present a survey of related work and an analysis of the data collected. We then describe the features used for our supervised learning approach to annotation cost estimation, followed by the experimental setup and results. Finally, we conclude with some future directions we would like to explore.

2 Related work

There has been some recent research effort in using supervised learning for estimating annotation cost. The most closely related work is that by Settles et al. (2008) and Ringger et al. (2008). Settles et al. (2008) present a detailed analysis of annotation cost for four NLP applications: named entity recognition, image retrieval, speculative vs. definite distinction, and information extraction. They study the effect of domain, annotator, jitter, order of examples, etc., on the annotation cost.

Results from Settles et al. (2008) are promising but leave much room for improvement. They used only instance level features such as number of entities, length, number of characters, percentage of non-alpha numeric characters, etc. for annotation cost estimation. For three of their tasks, the correlation between the estimated and actual annotation times was in the range ($R = 0.587$ to 0.852). Note that the percentage of variance accounted for by a model is obtained by squaring the R value from the correlation coefficient. Thus, an R value of 0.587 indicates that only about 34% (R^2) of the variance is accounted for, so the model will make incorrect predictions about ranking in the majority of cases. Nevertheless, we acknowledge that our results are not substantially better, although we argue that this work contributes to the pool of knowledge that will hopefully lead to better performance in the future.

Settles et al. (2008) train and test their estimator on data from the same annotator. Thus, in order to use their model for a new annotator, we would need to first collect data for that annotator and train a model. In our work, a group of annotators annotate the same text, and we train and test on different annotators. We also show that using characteristics of the annotators and annotation task in addition to the instance characteristics improves performance.

Ringger et al. (2008) use linear regression for annotation cost estimation for Part-Of-Speech (POS) tagging. About 30 annotators annotated 36 different instances each. The authors present about 13 descriptive statistics of the data, annotator and annotation task, but in their model they only used number of tokens in the sentence and the number of corrections needed as features. They report that the other variables didn't have a significant effect when evaluated using a Bayesian Information Criterion (from the R package).

Ringger et al. (2008) noticed that nativeness of the annotator did have an effect on the annotation time, but they chose not to include that feature in their model as they expected to have a similar mix of skills and background in their target annotators. However, if annotation times differ substantially across annotators, then not accounting for this difference will reduce the performance of the model. Also, the low adjusted correlation value for their model ($R = 0.181$) indicates that there is only a weak correlation between the annotation time and a linear combination of the length of the example and the number of corrections.

3 Analysis and Experiments

In this section, we present our annotation methodology and analysis of the data we collected, followed by a description of the features we used.We then present our experimental setup followed by a discussion of our results.

3.1 Annotation Methodology and Data Analysis

In this work, we estimate the annotation cost for a movie review classification task. The data we used were collected as part of a graduate course. Twenty annotators (students and instructors) were grouped into five groups of four each. The groups were created such that each group had similar variance in annotator characteristics such as department, educational experience, programming experience, etc. We used the first 200 movie reviews from the dataset provided by Zaidan et al. (2007), with an equal distribution of positive and negative examples. Each group annotated 25 movie reviews randomly selected from the 200 reviews and all annotators in each group annotated all 25 reviews. In addition to voting positive or negative for a review, annotators also annotated *rationales* (Zaidan et al., 2007), spans of text in the review that support their vote. Rationales can be used to guide the model by identifying the most discriminant features. In related work (Arora and Nyberg, 2009), we ascertain that with rationales the same performance can be achieved with less annotated data. The annotation task with rationales involved a variety of user actions: voting positive or negative, highlighting spans of text and adding rationale annotations. We used the same annotation guidelines as Zaidan et al. (2007). The data has been made available for research purposes[1]. Figure 1 shows a screenshot of the GUI used. We performed an analysis of our data similar to that conducted by Settles et al. (2008). We address the following main questions.

Are the annotation times variable enough? If all examples take a similar time to annotate, then the number of examples can be used as an approximation for the annotation cost. Figure 2 shows the histogram of averaged annotation times (averaged over 4 annotators in a group). As can be seen from the mean ($\mu = 165$ sec.) and the standard deviation ($\sigma = 68.85$), there is a meaningful variance in the annotation times.

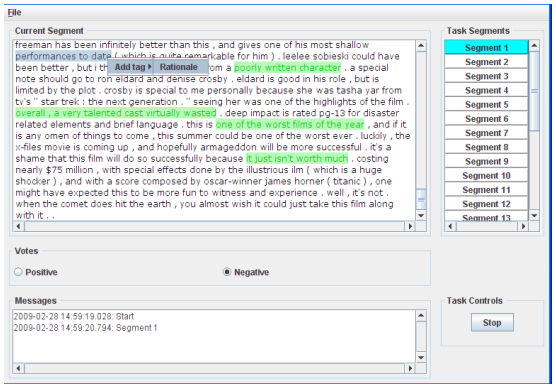


Figure 1: The GUI used for the annotation task. The user selects the review (segment) to annotate from the list in the right panel. The review text is displayed in the left panel. The user votes positive or negative using the radio buttons. Rationales are added by selecting a span of text and right clicking to select the rationale tag. The *start/stop* button can be used to pause the current task.

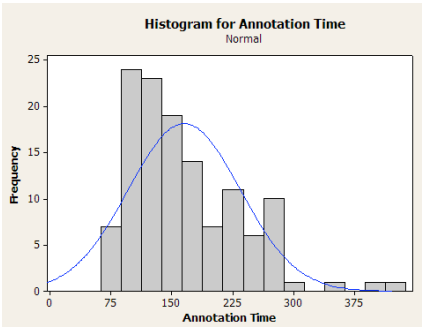


Figure 2: Distribution of averaged annotation times

How do the annotation times vary across annotators? A strong correlation between annotation times from different annotators on a set of instances suggests that there are certain characteristics of these instances, independent of the annotator characteristics, that can determine their ranking based on the time it takes to annotate them. We evaluated the pairwise correlation for all pairs of annotators in each group (Table 1). As can be seen, there is significant pairwise correlation in more than half of the pairs of annotators that differ in nativeness (10/16). However, not all such pairs of annotators are associated with significant correlation. This suggests that it is important to consider both instance and annotator characteristics for estimating annotation time.

[1] www.cs.cmu.edu/~shilpaa/datasets/ial/ial-uee-mr-v0.1.zip

group	Avg-Na(Std)	Avg-CR(Std)	#sign-pairs
0	2.25(0.96)	0.54(0.27)	4/6 (4/5)
1	1.75(0.5)	0.45(0.08)	5/6 (2/3)
2	1(0)	0.13(0.17)	0/6 (0/0)
3	1.75(0.96)	0.36(0.12)	2/6 (1/5)
4	2.75(0.5)	0.47(0.04)	6/6 (3/3)
Avg.	1.9(0.58)	0.39(0.21)	17/30 (10/16)

Table 1: The Table shows the average nativeness and average pairwise correlation between annotation times for the members of each group (and their standard deviation). #sign-pairs shows the fraction of pairwise correlations within the groups that were significant ($p < 0.05$). In brackets, is the fraction of correlations between annotators with different nativeness within the groups that were significant.

The box plot in Figure 3 shows the distribution of annotation times across annotators. As can be seen, some annotators take in general much longer than others, and the distribution of times is very different across annotators. For some, the annotation times vary a lot, but not so much for others. This suggests that using annotator characteristics as features in addition to the instance characteristics may be important for learning a better estimator.

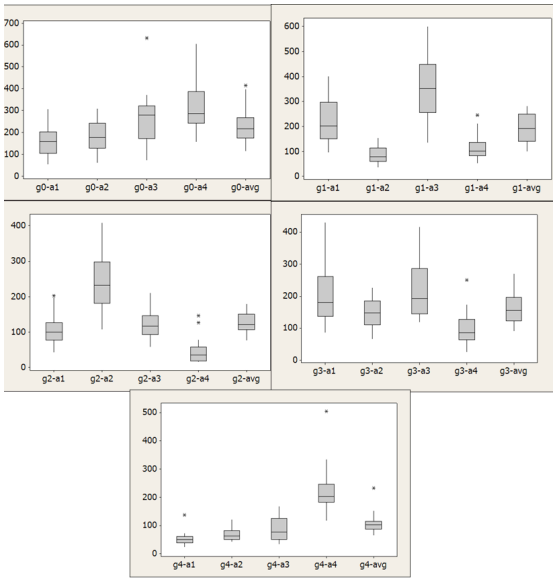


Figure 3: Box plot shows the annotation time (in sec) distribution (y-axis) for an annotator (x-axis) for a set of 25 documents. *g0-a1* represents annotator 1 of group 0 and *g0-avg* represents the average annotation time. A box represents the middle 50% of annotation times, with the line representing the median. Whiskers on either side span the 1^{st} and 4^{th} quartiles and asterisks indicate the outliers.

3.2 Feature Design

We group the features in the following three categories: Instance, Annotation Task and Annotator characteristics.

3.2.1 Instance characteristics

Instance characteristics capture the properties of the example the user annotates. Table 2 describes the instance based features we used and the intuition supporting their use for annotation cost estimation. Table 3 shows the mean and standard deviation of each of these characteristics, and as can be seen, these characteristics do vary across examples and hence these features can be beneficial for distinguishing examples.

3.2.2 Annotation Task characteristics

Annotation task characteristics are those that can be captured only during or after the annotation task. We used the number of rationales as a feature from this category. In addition to voting for movie reviews as positive or negative, the user also adds rationales that support their vote. More rationales imply more work since the user must look for the relevant span of text and perform the physical action of selecting the span and adding an annotation for each rationale. Table 3 shows the distribution of the average Number of Rationales (NR) per example (averaged over the four annotators for a given set).

3.2.3 Annotator characteristics

The annotation cost of an example may vary across annotators. As reported in Table 1, the average correlation for annotators on the same document is low ($R = 0.39$) with 17 out of 30 pairwise correlations being significant. Thus, it is important to consider annotator characteristics, such as whether the annotator is a native speaker of English, their education level, reading ability, etc. In this work, we only use nativeness of the annotator as a feature and plan to explore other characteristics in the future. We assigned each annotator a nativeness value. A value of 3 was given to an annotator whose first language is English. A value of 2 was given to an annotator who has a different first language but has either been educated in English or has been in the United States for a long time. A value of 1 was assigned to the remaining annotators. Among the 20 annotators in the study, there were 8 annotators with nativeness value of 1, and 6 each for nativeness values of 2 and 3. Table 1 shows the average and standard deviation of the nativeness score in each group.

Feature	Definition	Intuition
Character Length (CL)	Length of review in terms of number of characters	Longer documents take longer to annotate
Polar word Count (PC)	Number of words that are polar (strong subjective words from the lexicon (Wilson et al., 2005))	More subjectivity implies user would need more time to judge positive vs. negative
Stop word Percent (SC)	Percentage of words that are stop words	A high percentage of stop words implies that the text is not very complex and hence easier to read.
Avg. Sentence Length (SL)	Average of the character length of sentences in the review	Long sentences in a review may make it harder to read.

Table 2: Instance characteristics

Feature	Mean	Standard Deviation
CL	2.25	0.92
PC	41.50	20.39
SP	0.45	0.03
SL	121.90	28.72
NR	4.80	2.30

Table 3: Mean and the standard deviation for the feature occurrences in the data.

3.3 Evaluation Metric

We use both Root Mean Square (RMS) error and Correlation Coefficient (CRCoef) to evaluate our model, since the two metrics evaluate different aspects of an estimate. RMS is a way to quantify the amount by which an estimator differs from the true value of the quantity being estimated. It tells us how 'off' our estimate is from the truth. CRCoef on the other hand measures the strength and direction of a linear relationship between two random variables. It tells us how well correlated our estimate is with the actual annotation time. Thus, for evaluating how accurate our model is in predicting annotation times, RMS is a more appropriate metric. For evaluating the utility of the estimated annotation cost as a criterion for ranking and selecting examples for user's annotation, CRCoef is a better metric.

3.4 Experiments & Results

We learn an annotation cost estimator using the Linear Regression and SMO Regression (Smola and Scholkopf, 1998) learners from the Weka machine learning toolkit (Witten and Frank, 2005). As mentioned earlier, we have 5 sets of 25 documents each, and each set was annotated by four annotators. The results reported are averaged over five folds, where each set is one fold, and two algorithms (Linear Regression and SMO Regression). Varying the algorithm helps us find the most predictive feature combinations across different algorithms. Since each set was annotated by different annotators, we never train and test on the data from same annotators. We used the JMP[2] and Minitab[3] statistical tools for our analysis. We used an ANOVA model with Standard Least Squares fitting to compare the different experimental conditions. We make all comparisons in terms of both the CRCoef and the RMS metrics. For significance results reported, we used 2-tailed paired T-test, considering ($p < 0.05$) as significant.

We present our results and analysis in three parts. We first compare the four instance characteristics, annotator and annotation task characteristics; and their combination. We then present an analysis of the interaction between features and annotation time. Finally, we compare the ranking of features based on the two evaluation metrics we used.

3.4.1 Comparing characteristics for annotation cost estimation

Instance Characteristics: We compare the four instance characteristics described in Section 3.2.1 and select the most predictive characteristic for further analysis with annotator and annotation task characteristics. As can be seen in Table 4, character length performs the best, and it is significantly better than stop word percent and average sentence length. Character length also outperforms polar word count, but this difference is not significant. Because of the large significant difference between the performance of stop word percent and average sentence length, compared to character length, we do not consider them for further analysis.

Feature Combinations: In Table 5, we compare the feature combinations of instance, annotator and annotation task characteristics. The table also shows the weights for the features used and the constant for the linear regression model trained on all the data. A missing weight for a feature indicates that it wasn't used in that feature combination.

[2] http://www.jmp.com/software/

[3] http://www.minitab.com/

Feature	CR-Coef	RMS
CL	**0.358**	**104.51**
PC	0.337	105.92
SP	-0.041*	114.34*
SL	0.042*	114.50*

Table 4: CR-Coef and RMS results for Character Length (CL), Polar word Count (PC), Stop word Percent (SP) and average Sentence Length (SL). Best performance is highlighted in bold. * marks the results significantly worse than the best.

We use only the best performing instance characteristic, the character length. The length of an example has often been substituted for the annotation cost (Kapoor et al., 2007; Tomanek et al., 2007). We show in Table 5 that certain feature combinations significantly outperform character length. The combination of all three features (last row) performs the best for both CRCoef and RMS, and this result is significantly better than the character length (third row). The combination of number of rationales and nativeness (fourth row) also outperforms character length significantly in CRCoef. This suggests that the number of rationales we expect or require in a review and the annotator characteristics are important factors for annotation cost estimation and should be considered in addition to the character length.

CL	NR	AN	Const.	CR-Coef	RMS
		-29.33	220.77	0.135*	123.93*
	17.59		82.81	0.486	95.29
0.027			61.53	0.357*	104.51*
	19.11	-40.78	153.21	0.55+	96.04
0.028		32.79	120.18	0.397*	109.85*
0.02	15.15		17.57	0.553+	90.27+
0.021	16.64	-41.84	88.09	**0.626**+	**88.44**+

Table 5: CR-Coef and RMS results for seven feature combinations of Character Length (CL), Number of Rationales (NR) and Annotator Nativeness (AN). The values in feature and 'Const.' columns are weights and constant for the linear regression model trained on all the data. The numbers in bold are the results for the best feature combination. * marks the results significantly worse than the best. + marks the results significantly better than CL.

The impact of the nativeness feature is somewhat mixed. Adding the nativeness feature always improves the correlation and for RMS, it helps when added to the combined feature (CL+NR) but not otherwise. Although this improvement with addition of the nativeness feature is not significant, it does suggest that annotator characteristics might be important to consider. To investigate this further, we evaluated our assumption that native speakers take less time to annotate. For each set, we compared the average annotation times (averaged over examples) against the nativeness values. For all sets, annotators with nativeness value of 3 always took less time on average than those with nativeness value of 2 or 1. Between 2 and 1, there were no reliable differences. Sometimes annotators with value of 1 took less time than annotators with value of 2. Also, for group 2 which had all annotators with nativeness value of 1, we observed a poor correlation between annotators (Table 1). This suggest two things: 1) our assignment of nativeness value may not be accurate and we need other ways of quantifying nativeness, 2) there are other annotator characteristics we should take into consideration.

PC	CL	NR	AN	Const.	CR	RMS
	0.027			61.53	**0.358**ab	**104.5x**
2.2				74.20	0.337a	105.9x
0.7	0.019			60.89	0.355b	104.9x
	0.028		-32.8	120.2	**0.397**ab	**109.8x**
2.3			-35.5	135.1	0.382a	111.1x
1.1	0.016		-34.3	121.8	0.395b	109.9x
	0.02	15.1		17.57	0.553a	**90.27x**
1.5		15.1		32.02	0.542a	91.65x
0.0	0.02	15.1		17.57	**0.554**a	90.40x
	0.021	16.6	-41.8	88.09	**0.626**a	**88.44x**
1.6		16.5	-43.5	102.8	0.614a	90.42y
0.0	0.021	16.6	-41.8	88.09	**0.626**a	88.78x

Table 6: Each block of 3 rows in this table compares the performance of Character Length (CL) and Polar word Count (PC) in combination with Number of Rationales (NR) and Annotator Nativeness (AN) features. The values in feature and 'Const.' columns are weights and constant for the linear regression model trained on all the data. Best performance is highlighted in bold. Results in a block not connected by same letter are significantly different.

Polar word Count and Character Length: As we saw in Table 4, the difference between character length and polar word count is not significant. We further compare these two instance characteristics in the presence of the annotator and annotation task characteristics. Our goal is to ascertain whether character length performs better than polar word count, or vice versa, and whether this difference is significant. We also evaluate whether using both performs better than using any one of them alone. The results presented in Table 6 help us answer these questions. For all feature combinations character length, with and without polar word count, performs

better than polar word count, but this difference is not significant except in three cases. These results suggests that polar word count can be used as an alternative to character length in annotation cost estimation.

3.4.2 Interaction between Features and Annotation Time

As a post-experiment analysis, we studied the interaction between the features we used and annotation time, and the interaction among features themselves. Table 7 reports the pairwise correlation (Pearson, 1895) for these variables, calculated over all 125 reviews. As can be seen, all features have significant correlation with annotation time except stop words percentage and average sentence length.

Note that number of rationales has higher correlation with annotation time ($R = 0.529$) than character length ($R = 0.417$). This suggests that number of rationales may have more influence than character length on annotation time, and a low correlation between number of rationales and character length ($R = 0.238$) indicates that it might not be the case that longer documents necessarily contain more rationales. Annotating rationales requires cognitive effort of identifying the right span and manual effort to highlight and add an annotation, and hence more rationales implies more annotation time. We also found some examples in our data where documents with substantially different lengths but same number of rationales took a similar time to annotate. One possible explanation for this observation is user's annotation strategy. If the annotator chooses to skim through the remaining text when enough rationales are found, two examples with same number of rationales but different lengths might take similar time. We plan to investigate the effect of annotator's strategy on annotation time in the future.

A negative correlation of nativeness with annotation time ($R = -0.219$) is expected, since native speakers ($AN = 3$) are expected to take less annotation time than non-native speakers ($AN = \{2, 1\}$), although this correlation is low. A low correlation between number of rationales and nativeness ($R = 0.149$) suggests that number of rationales a user adds may not be influenced much by their nativeness value. A not significant low correlation ($R = -0.06$) between character length and nativeness provides no evidence that reviews with different lengths were distributed non-uniformly across annotators with different nativeness.

	AT	CL	NR	AN	PC	SP	SL
AT	1						
CL	**0.42**	1					
NR	**0.53**	**0.24**	1				
AN	**-0.22**	0.06	**0.15**	1			
PC	**0.4**	**0.89**	**0.28**	**0.11**	1		
SP	0.03	0.06	**0.14**	0.03	0.04	1	
SL	0.08	**0.15**	0.01	-0.01	**0.14**	**-0.13**	1

Table 7: Correlation between Character Length (CL), Number of Rationales (NR), Annotator Nativeness (AN), Polar word Count (PC), Stop word Percent (SP), average Sentence Length (SL) and Annotation Time (AT), calculated over all documents (125) and all annotators (20). Significant correlations are highlighted in bold.

The number of polar words in a document has a similar correlation with annotation time as character length ($R = 0.4$). There is also a strong correlation between character length and polar word count ($R = 0.89$). Since reviews are essentially people's opinions, we can expect longer documents to have more polar words. This also explains why there is no significant difference in performance for polar word count and character length (Table 4). A more useful feature may be the information about the number of positive and negative polar words in a review, since a review with both positive and negative opinions can be difficult to classify as positive or negative. We plan to explore these variations of the polar word feature in the future. We also plan to investigate how we can exploit this dependence between characteristics for annotation cost estimation.

3.4.3 CRCoef Vs. RMS

We presented our results using correlation coefficient and root mean squared error metrics. Table 8 shows the ranking of the feature combinations from better to worse for both these metrics and as we can see, there is a difference in the order of feature combinations for the two metrics. Also, significance results differ in some cases for the two metrics. These differences suggest that features which correlate well with the annotation times (higher CRCoef rank) can give an accurate ranking of examples based on their annotation cost, but they may not be as accurate in their absolute estimate for simulating annotators and thus might have a lower RMS rank. Thus, it is important to evaluate the user effort esti-

mator in terms of both these metrics so that the right estimator can be chosen for a given objective.

Rank	CR-Coef	RMS
1	(CL+NR+AN)	(CL+NR+AN)
2	(CL+NR)	(CL+NR)
3	(NR+AN)	(NR)
4	(NR)	(NR+AN)
5	(CL+AN)	(CL)
6	(CL)	(CL+AN)
7	(AN)	(AN)

Table 8: Ranking of feature combinations.

4 Towards a General Annotation Cost Estimator

Our multi-annotator environment allows us to train and test on data from different annotators by using annotator characteristics as features in the annotation cost estimation. A model trained on data from a variety of annotators can be used for recommending examples to annotators not represented in our training data but with similar characteristics. This is important since we may not always know all our annotators before building the model, and training an estimator for each new annotator is costly. Also, in active learning research, the goal is to evaluate selective sampling approaches independently of the annotator. Choosing annotators for supervised annotation cost estimation such that the within group variance in annotator characteristics is high will give us a more generic estimator and a stricter evaluation criterion. Thus, we have a framework that has the potential to be used to build a user-independent annotation cost estimator for a given task.

However, this framework is specific to the User Interface (UI) used. A change in the user interface might require recollecting the data from all the annotators and training a model on the new data. For example, if annotating rationales was made significantly faster in a new UI design, it would have a major impact on annotation cost. An alternative would be to incorporate UI features in our model and train it on several different UIs or modifications of the same UI, which will allow us to use our trained model with a new user interface or modifications of the existing UIs, without having to recollect the data and retrain the model. A few UI features that can be used in our context are: adding a rationale annotation, voting positive or negative, etc. The units for expressing these features will be the low-level user interface actions such as number of clicks, mouse drags, etc. For example, in our task, adding a rationale annotation requires one mouse drag and two clicks, and adding a vote requires one click. In a different user interface, adding a rationale annotation might require just one mouse drag.

Using UI features raises a question of whether they can replace the annotation task features; e.g., whether the UI feature for adding rationale annotation can replace the number of rationales feature. Our hypothesis is that number of rationales has more influence on annotation time than just the manual effort of annotating them. It also requires the cognitive effort of finding the rationale, deciding its span, etc. We aim to explore incorporating UI features in our annotation cost estimation model in the future.

5 Conclusion and Future Work

In this work we presented a detailed investigation of annotation cost estimation for active learning with multiple annotators. We motivated the task from two perspectives: selecting examples to minimize annotation cost and simulating annotators for evaluating active learning approaches. We defined three categories of features based on instance, annotation task and annotator characteristics. Our results show that using a combination of features from all three categories performs better than any one of them alone. Our analysis was limited to a small dataset. In the future, we plan to collect a larger dataset for this task and explore more features from each feature group.

With the multi-annotator annotation cost estimator proposed, we also motivated the need for a general estimator that can be used with new annotators or user interfaces without having to retrain. We aim to explore this direction in the future by extending our model to incorporate user interface features. We also plan to use the annotation cost model we developed in an active learning experiment.

Acknowledgments

We would like to thank Hideki Shima for his help with the task setup and Jing Yang for helpful discussions. We would also like to thank all the anonymous reviewers for their helpful comments.

References

Shilpa Arora and Eric Nyberg. 2009. *Interactive Annotation Learning with Indirect Feature Voting*. In Proceedings of NAACL-HLT 2009 (Student Research Workshop).

Xavier Carreras and Lluìs Márquez. 2004. *Introduction to the CoNLL-2004 Shared Task: Semantic Role Labeling*. `http://www.lsi.upc.edu/~srlconll/st04/st04.html`.

Gahgene Gweon, Carolyn Penstein Ros'e, Joerg Wittwer and Matthias Nueckles. 2005. *Supporting Efficient and Reliable Content Analysis Using Automatic Text Processing Technology*. In proceedings of INTERACT 2005: 1112-1115.

Robbie A. Haertel, Kevin D. Seppi, Eric K. Ringger and Janes L. Cattoll. 2008. *Return on Investment for Active Learning*. In proceedings of NIPS Workshop on Cost Sensitive Learning.

Rebecca Hwa. 2000. *Sample Selection for Statistical Grammar Induction*. In proceedings of joint SIGDAT conference on Empirical Methods in NLP and Very Large Corpora.

Ashish Kapoor, Eric Horvitz and Sumit Basu. 2007. *Selective supervision:Guiding supervised learning with decision-theoretic active learning*. In proceedings of IJCAI, pages 877-882.

Ross D. King, Kenneth E. Whelan, Ffion M. Jones, Philip G. K. Reiser, Christopher H. Bryant, Stephen H. Muggleton, Douglas B. Kell and Stephen G. Oliver. 2004. *Functional Genomics hypothesis generation and experimentation by a robot scientist*. In proceedings of Nature, 427(6971):247-52.

Trausti Kristjansson, Aron Culotta, Paul Viola and Andrew Mccallum. 2004. *Interactive Information Extraction with Constrained Conditional Random Fields*. In proceedings of AAAI.

Karl Pearson. 1895. *Correlation Coefficient*. Royal Society Proceedings, 58, 214.

Eric Ringger, Marc Carmen, Robbie Haertel, Kevin Seppi, Deryle Lonsdale, Peter McClanahan, Janes L. Cattoll and Noel Ellison. 2008. *Assessing the Costs of Machine-Assisted Corpus Annotation through a User Study*. In proceedings of LREC.

Burr Settles, Mark Craven and Lewis Friedland. 2008. *Active Learning with Real Annotation Costs*. In proceedings of NIPS Workshop on Cost Sensitive Learning.

Alex J. Smola and Bernhard Scholkopf 1998. *A Tutorial on Support Vector Regression*. NeuroCOLT2 Technical Report Series - NC2-TR-1998-030.

Katrin Tomanek, Joachim Wermter and Udo Hahn. 2007. *An approach to text corpus construction which cuts annotation costs and maintains reusability of annotated data*. In proceedings of EMNLP-CoNLL, pp. 486-495.

Theresa Wilson, Janyce Wiebe and Paul Hoffmann. 2005. *Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis*. In proceedings of HLT/EMNLP, Vancouver, Canada.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. 2nd Edition, Morgan Kaufmann, San Francisco.

Omar Zaidan, Jason Eisner and Christine Piatko. 2007. *Using "annotator rationales" to improve machine learning for text categorization*. In Proceedings of NAACL-HLT, pp. 260-267, Rochester, NY.

Data Quality from Crowdsourcing: A Study of Annotation Selection Criteria

Pei-Yun Hsueh, Prem Melville, Vikas Sindhwani
IBM T.J. Watson Research Center
1101 Kitchawan Road, Route 134
Yorktown Heights, NY 10598, USA

Abstract

Annotation acquisition is an essential step in training supervised classifiers. However, manual annotation is often time-consuming and expensive. The possibility of recruiting annotators through Internet services (e.g., Amazon Mechanic Turk) is an appealing option that allows multiple labeling tasks to be outsourced in bulk, typically with low overall costs and fast completion rates. In this paper, we consider the difficult problem of classifying sentiment in political blog snippets. Annotation data from both expert annotators in a research lab and non-expert annotators recruited from the Internet are examined. Three selection criteria are identified to select high-quality annotations: noise level, sentiment ambiguity, and lexical uncertainty. Analysis confirm the utility of these criteria on improving data quality. We conduct an empirical study to examine the effect of noisy annotations on the performance of sentiment classification models, and evaluate the utility of annotation selection on classification accuracy and efficiency.

1 Introduction

Crowdsourcing (Howe, 2008) is an attractive solution to the problem of cheaply and quickly acquiring annotations for the purposes of constructing all kinds of predictive models. To sense the potential of crowdsourcing, consider an observation in von Ahn et al. (2004): a crowd of 5,000 people playing an appropriately designed computer game 24 hours a day, could be made to label all images on Google (425,000,000 images in 2005) in a matter of just 31 days. Several recent papers have studied the use of annotations obtained from Amazon Mechanical Turk, a marketplace for recruiting online workers (Su et al., 2007; Kaisser et al., 2008; Kittur et al., 2008; Sheng et al., 2008; Snow et al., 2008; Sorokin and Forsyth, 2008).

With efficiency and cost-effectiveness, online recruitment of anonymous annotators brings a new set of issues to the table. These workers are not usually specifically trained for annotation, and might not be highly invested in producing good-quality annotations. Consequently, the obtained annotations may be noisy by nature, and might require additional validation or scrutiny. Several interesting questions immediately arise in how to optimally utilize annotations in this setting: How does one handle differences among workers in terms of the quality of annotations they provide? How useful are noisy annotations for the end task of creating a model? Is it possible to identify genuinely ambiguous examples via annotator disagreements? How should these considerations be treated with respect to intrinsic informativeness of examples? These questions also hint at a strong connection to active learning, with annotation quality as a new dimension to the problem.

As a challenging empirical testbed for these issues, we consider the problem of sentiment classification on political blogs. Given a snippet drawn from a political blog post, the desired output is a polarity score that indicates whether the sentiment expressed is positive or negative. Such an analysis provides a view of the opinion around a subject of interest, e.g., US Presidential candidates, aggregated across the blogsphere. Recently, sentiment analy-

Proceedings of the NAACL HLT Workshop on Active Learning for Natural Language Processing, pages 27–35,
Boulder, Colorado, June 2009.

sis is emerging as a critical methodology for social media analytics. Previous research has focused on classifying subjective-versus-objective expressions (Wiebe et al., 2004), and also on accurate sentiment polarity assignment (Turney, 2002; Yi et al., 2003; Pang and Lee, 2004; Sindhwani and Melville, 2008; Melville et al., 2009).

The success of most prior work relies on the quality of their knowledge bases; either lexicons defining the sentiment polarity of words around a topic (Yi et al., 2003), or quality annotation data for statistical training. While manual intervention for compiling lexicons has been significantly lessened by bootstrapping techniques (Yu and Hatzivassiloglou, 2003; Wiebe and Riloff, 2005), manual intervention in the annotation process is harder to avoid. Moreover, the task of annotating blog-post snippets is challenging, particularly in a charged political atmosphere with complex discourse spanning many issues, use of cynicism and sarcasm, and highly domain-specific and contextual cues. The downside is that high-performance models are generally difficult to construct, but the upside is that annotation and data-quality issues are more clearly exposed.

In this paper we aim to provide an empirical basis for the use of data selection criteria in the context of sentiment analysis in political blogs. Specifically, we highlight the need for a set of criteria that can be applied to screen untrustworthy annotators and select informative yet unambiguous examples for the end goal of predictive modeling. In Section 2, we first examine annotation data obtained by both the expert and non-expert annotators to quantify the impact of including non-experts. Then, in Section 3, we quantify criteria that can be used to select annotators and examples for selective sampling. Next, in Section 4, we address the questions of whether the noisy annotations are still useful for this task and study the effect of the different selection criteria on the performance of this task. Finally, in Section 5 we present conclusion and future work.

2 Annotating Blog Sentiment

This section introduces the Political Blog Snippet (PBS) corpus, describes our annotation procedure and the sources of noise, and gives an overview of the experiments on political snippet sentiments.

2.1 The Political Blog Snippet Corpus

Our dataset comprises of a collection of snippets extracted from over 500,000 blog posts, spanning the activity of 16,741 political bloggers in the time period of Aug 15, 2008 to the election day Nov 4, 2008. A snippet was defined as a window of text containing four consecutive sentences such that the head sentence contained either the term "Obama" or the term "McCain", but both candidates were not mentioned in the same window. The global discourse structure of a typical political blog post can be highly complicated with latent topics ranging from policies (e.g., financial situation, economics, the Iraq war) to personalities to voting preferences. We therefore expected sentiment to be highly non-uniform over a blog post. This snippetization procedure attempts to localize the text around a presidential candidate with the objective of better estimating aggregate sentiment around them. In all, we extracted 631,224 snippets. For learning classifiers, we passed the snippets through a stopword filter, pruned all words that occur in less than 3 snippets and created normalized term-frequency feature vectors over a vocabulary of 3,812 words.

2.2 Annotation Procedure

The annotation process consists of two steps:

Sentiment-class annotation: In the first step, as we are only interested in detecting sentiments related to the named candidate, the annotators were first asked to mark up the snippets irrelevant to the named candidate's election campaign. Then, the annotators were instructed to tag each relevant snippet with one of the following four sentiment polarity labels: Positive, Negative, Both, or Neutral.

Alignment annotation: In the second step, the annotators were instructed to mark up whether each snippet was written to support or oppose the target candidate therein named. The motivation of adding this tag comes from our interest in building a classification system to detect positive and negative mentions of each candidate. For the snippets that do not contain a clear political alignment, the annotators had the freedom to mark it as neutral or simply not alignment-revealing.

In our pilot study many bloggers were observed to endorse a named candidate by using negative ex-

pressions to denounce his opponent. Therefore, in our annotation procedure, the distinction is made between the coding of manifest content, i.e., sentiments "on the surface", and latent political alignment under these surface elements.

2.3 Agreement Study

In this section, we compare the annotations obtained from the on-site expert annotators and those from the non-expert AMT annotators.

2.3.1 Expert (On-site) Annotation

To assess the reliability of the sentiment annotation procedure, we conducted an agreement study with three expert annotators in our site, using 36 snippets randomly chosen from the PBS Corpus. Overall agreement among the three annotators on the relevance of snippets is 77.8%. Overall agreement on the four-class sentiment codings is 70.4%.

Analysis indicate that the annotators agreed better on some codings than the others. For the task of determining whether a snippet is subjective or not[1], the annotators agreed 86.1% of the time. For the task of determining whether a snippet is positive or negative, they agreed 94.9% of the time.

To examine which pair of codings is the most difficult to distinguish, Table 1 summarizes the confusion matrix for the three pairs of annotator's judgements on sentiment codings. Each column describes the marginal probability of a coding and the probability distribution for this coding being recognized as another coding (including itself). As many bloggers use cynical expressions in their writings, the most confusing cases occur when the annotators have to determine whether a snippet is "negative" or "neutral". The effect of cynical expressions on sentiment analysis in the political domain is also revealed in the second step of alignment annotation. Only 42.5% of the snippets have been coded with alignment coding in the same direction as its sentiment coding – i.e., if a snippet is intended to support (oppose) a target candidate, it will contain positive (negative) sentiment. The alignment coding task has been shown to be reliable, with the annotators agreeing 76.8% of the time overall on the three-level codings: Support/Against/Neutral.

%	Neu	Pos	Both	Neg
Marginal	21.9	20.0	10.5	47.6
Neutral (Neu)	47.8	14.3	9.1	16.0
Positive (Pos)	13.0	61.9	18.2	6.0
Both (Both)	4.4	9.5	9.1	14.0
Negative (Neg)	34.8	14.3	63.6	64.0

Table 1: Summary matrix for the three on-site annotators' sentiment codings.

[1]This is done by grouping the codings of Positive, Negative, and Both into the subjective class.

2.3.2 Amazon Mechanical Turk Annotation

To compare the annotation reliability between expert and non-expert annotators, we further conducted an agreement study with the annotators recruited from Amazon Mechanical Turk (AMT). We have collected 1,000 snippets overnight, with the cost of 4 cents per annotation.

In the agreement study, a subset of 100 snippets is used, and each snippet is annotated by five AMT annotators. These annotations were completed by 25 annotators whom were selected based on the approval rate of their previous AMT tasks (over 95% of times).[2] The AMT annotators spent on average 40 seconds per snippet, shorter than the average of two minutes reported by the on-site annotators. The lower overall agreement on all four-class sentiment codings, 35.3%, conforms to the expectation that the non-expert annotators are less reliable. The Turk annotators also agreed less on the three-level alignment codings, achieving only 47.2% of agreement.

However, a finer-grained analysis reveals that they still agree well on some codings: The overall agreement on whether a snippet is relevant, whether a snippet is subjective or not, and whether a snippet is positive or negative remain within a reasonable range: 81.0%, 81.8% and 61.9% respectively.

2.4 Gold Standard

We defined the gold standard (GS) label of a snippet in terms of the coding that receives the majority votes.[3] Column 1 in Table 2 (onsite-GS predic-

[2]Note that we do not enforce these snippets to be annotated by the same group of annotators. However, $Kappa$ statistics requires to compute the chance agreement of each annotator. Due to the violation of this assumption, we do not measure the intercoder agreement with $Kappa$ in this agreement study.

[3]In this study, we excluded 6 snippets whose annotations failed to reach majority vote by the three onsite annotators.

	onsite-GS prediction	onsite agreement		AMT-GS prediction	AMT agreement
Sentiment (4-class)	0.767	0.704		0.614	0.353
Alignment (3-level)	0.884	0.768		0.669	0.472
Relevant or not	0.889	0.778		0.893	0.810
Subjective or not	0.931	0.861		0.898	0.818
Positive or negative	0.974	0.949		0.714	0.619

Table 2: Average prediction accuracy on gold standard (GS) using one-coder strategy and inter-coder agreement.

tion) shows the ratio of the onsite expert annotations that are consistent with the gold standard, and Column 3 (AMT-GS prediction) shows the same for the AMT annotations. The level of consistency, i.e., the percentage agreement with the gold standard labels, can be viewed as a proxy of the quality of the annotations. Among the AMT annotations, Columns 2 (onsite agreement) and 4 (AMT agreement) show the pair-wise intercoder agreement in the on-site expert and AMT annotations respectively.

The results suggest that it is possible to take one single expert annotator's coding as the gold standard in a number of annotation tasks using binary classification. For example, there is a 97.4% chance that one expert's coding on the polarity of a snippet, i.e., whether it is positive or negative, will be consistent with the gold standard coding. However, this one-annotator strategy is less reliable with the introduction of non-expert annotators. Take the task of polarity annotation as an example, the intercoder agreement among the AMT workers goes down to 61.9% and the "one-coder" strategy can only yield 71.4% accuracy. To determine reliable gold standard codings, multiple annotators are still necessary when non-expert annotators are recruited.

3 Annotation Quality Measures

Given the noisy AMT annotations, in this section we discuss some summary statistics that are needed to control the quality of annotations.

3.1 Annotator-level noise

To study the question of whether there exists a group of annotators who tend to yield more noisy annotations, we evaluate the accumulated noise level introduced by each of the annotators. We define the noise level as the deviation from the gold standard labels. Similar to the measure of individual error rates proposed in (Dawid and Skene, 1979), the noise level of a particular annotator j, i.e., $noise(anno_j)$, is then estimated by summing up the deviation of the annotations received from this annotator, with a small sampling correction for chance disagreement. Analysis results demonstrate that there does exist a subset of annotators who yield more noisy annotations than the others. 20% of the annotators (who exceed the noise level 60%) result in annotations that have 70% disagreement with the gold standard.

In addition, we also evaluate how inclusion of noisy annotators reduces the mean agreement with Gold Standard. The plot (left) in Figure 1 plots the mean agreement rate with GS over the subset of annotators that pass a noise threshold. These results show that the data quality decreases with the inclusion of more untrustworthy annotators.

3.2 Snippet-level sentiment ambiguity

We have observed that not all snippets are equally easy to annotate, with some containing more ambiguous expressions. To incorporate this concern in the selection process, a key question to be answered is whether there exist snippets whose sentiment is substantially less distinguishable than the others.

We address this question by quantifying ambiguity measures with the two key properties shown as important in evaluating the controversiality of annotation snippets (Carenini and Cheung, 2008): (1) the strength of the annotators' judgements and (2) the polarity of the annotations. The measurement needs to satisfy the constraints demonstrated in the following snippets: (1) An example that has received three positive codings are more ambiguous than that has received five, and (2) an example that has received five positive codings is more ambiguous than the one that has received four positive and one negative coding. In addition, as some snippets were shown to

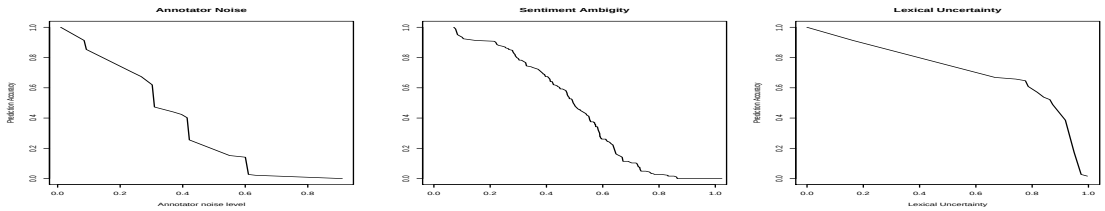


Figure 1: *Data quality (consistency with GS) as a function of noise level (left), sentiment ambiguity (middle), and lexical uncertainty (right).*

be difficult to tell whether they contain negative or neutral sentiment, the measure of example ambiguity has to go beyond controversiality and incorporate codings of "neutral" and "both".

To satisfy these constraints, we first enumerated through the codings of each snippet and counted the number of neutral, positive, both, and negative codings: We added (1) one to the positive (negative) category for each positive (negative) coding, (2) 0.5 to the neutral category with each neutral coding, and (3) 0.5 to both the positive and negative categories with each both coding. The strength of codings in the three categories, i.e., $str_+(snip_i)$, $str_{neu}(snip_i)$, and $str_-(snip_i)$, were then summed up into $str(snip_i)$. The distribution were parameterized with

$$\theta_+(snip_i) = str_+(snip_i)/str(snip_i)$$
$$\theta_{neu}(snip_i) = str_{neu}(snip_i)/str(snip_i)$$
$$\theta_-(snip_i) = str_-(snip_i)/str(snip_i)$$

We then quantify the level of ambiguity in the annotator's judgement as follows:

$$H(\theta(snip_i)) = -\theta_+(snip_i)log(\theta_+(snip_i)) - \theta_{neu}(snip_i)log(\theta_{neu}(snip_i)) - \theta_-(snip_i)log(\theta_-(snip_i))$$

$$Amb(snip_i) = \frac{str(snip_i)}{str_{max}} \times H(\theta(snip_i)),$$

where str_{max} is the maximum value of str among all the snippets in the collection. The plot (middle) in Figure 1 shows that with the inclusion of snippets that are more ambiguous in sentiment disambiguation, the mean agreement with Gold Standard decreases as expected.

3.3 Combining measures on multiple annotations

Having established the impact of noise and sentiment ambiguity on annotation quality, we then set out to explore how to integrate them for selection. First, the ambiguity scores for each of the snippets are reweighed with respect to the noise level.

$$w(snip_i) = \sum_j noise(anno_j) \times (\frac{1}{e})^{\theta(ij)}$$
$$Conf(snip_i) = \frac{w(snip_i)}{\sum_i w(snip_i)} \times Amb(snip_i),$$

where $\theta(ij)$ is an indicator function of whether a coding of $snip_i$ from annotator j agrees with its gold standard coding. $w(exp_i)$ is thus computed as the aggregated noise level of all the annotators who labeled the ith snippet.

To understand the baseline performance of the selection procedure, we evaluate the the true predictions versus the false alarms resulting from using each of the quality measures separately to select annotations for label predictions. In this context, a true prediction occurs when an annotation suggested by our measure as high-quality indeed matches the GS label, and a false alarm occurs when a high quality annotation suggested by our measure does not match the GS label. The ROC (receiver operating characteristics) curves in Figure 2 reflect all the potential operating points with the different measures.

We used data from 2,895 AMT annotations on 579 snippets, including 63 snippets used in the agreement study. This dataset is obtained by filtering out the snippets with their GS labels as 1 ("irrelevant") and the snippets that do not receive any coding that has more than two votes.

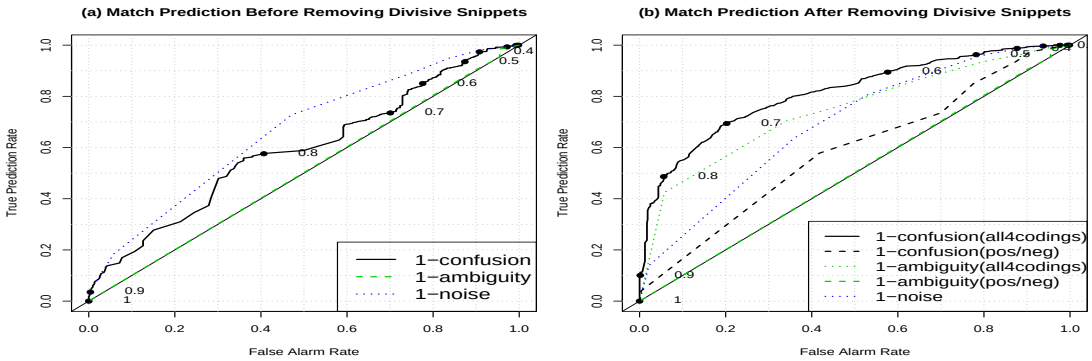


Figure 2: *Modified ROC curves for quality measures: (a) before removing divisive snippets, (b) after removing divisive snippets. The numbers shown with the ROC curve are the values of the aggregated quality measure (1-confusion).*

Initially, three quality measures are tested: 1-noise, 1-ambiguity, 1-confusion. Examination of the snippet-level sentiment codings reveals that some snippets (12%) result in "divisive" codings, i.e., equal number of votes on two codings.

The ROC curves in Figure 2 (a) plot the baseline performance of the different quality measures. Results show that before removing the subset of divisive snippets, the only effective selection criteria is obtained by monitoring the noise level of annotators. Figure 2 (b) plots the performance after removing the divisive snippets. In addition, our ambiguity scores are computed under two settings: (1) with only the polar codings (pos/neg), and (2) with all the four codings (all4codings). The ROC curves reveal that analyzing only the polar codings is not sufficient for annotation selection.

The results also demonstrate that confusion, an integrated measure, does perform best. Confusion is just one way of combining these measures. One may chose alternative combinations – the results here primarily illustrate the benefit of considering these different dimensions in tandem. Moreover, the difference between plot (a) and (b) suggests that removing divisive snippets is essential for the quality measures to work well. How to automatically identify the divisive snippets is therefore important to the success of the annotation selection process.

3.4 Effect of lexical uncertainty on divisive snippet detection

In search of measures that can help identify the divisive snippets automatically, we consider the inherent lexical uncertainty of an example. Uncertainty Sampling (Lewis and Catlett, 1994) is one common heuristic for the selection of informative instances, which select instances that the current classifier is most uncertain about. Following on these lines we measure the uncertainty on instances, with the assumption that the most uncertain snippets are likely to be divisive.

In particular, we applied a lexical sentiment classifier (c.f. Section 4.1.1) to estimate the likelihood of an unseen snippet being of positive or negative sentiment, i.e., $P_+(exp_i)$, $P_-(exp_i)$, by counting the sentiment-indicative word occurrences in the snippet. As in our dataset the negative snippets far exceed the positive ones, we also take the prior probability into account to avoid class bias. We then measure lexical uncertainty as follows.

$$\begin{aligned} &Deviation(snip_i) = \\ &\frac{1}{C} \times |(log(P(+)) - log(P(-))) \\ &+(log(P_+(snip_i)) - log(P_-(snip_i)))|, \\ &Uncertainty(snip_i) = 1 - Deviation(snip_i), \end{aligned}$$

where class priors, $P(+)$ and $P(-)$, are estimated with the dataset used in the agreement studies, and C is the normalization constant.

We then examine not only the utility of lexical uncertainty in identifying high-quality annotations, but

Classifier	Accuracy	AUC
LC	49.60	0.614
NB	83.53	0.653
SVM	83.89	0.647
Pooling	84.51	0.700

Table 3: Accuracy of sentiment classification methods.

also the utility of such measure in identifying divisive snippets. Figure 1 (right) shows the effect of lexical uncertainty on filtering out low-quality annotations. Figure 3 demonstrates the effect of lexical uncertainty on divisive snippet detection, suggesting the potential use of lexical uncertainty measures in the selection process.

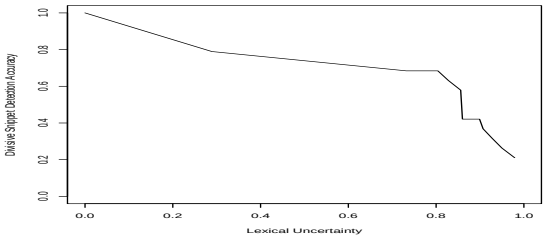


Figure 3: *Divisive snippet detection accuracy as a function of lexical uncertainty.*

4 Empirical Evaluation

The analysis in Sec. 3 raises two important questions: (1) how useful are noisy annotations for sentiment analysis, and (2) what is the effect of online annotation selection on improving sentiment polarity classification?

4.1 Polarity Classifier with Noisy Annotations

To answer the first question raised above, we train classifiers based on the noisy AMT annotations to classify positive and negative snippets. Four different types of classifiers are used: SVMs, Naive Bayes (NB), a lexical classifier (LC), and the lexical knowledge-enhanced Pooling Multinomial classifier, described below.

4.1.1 Lexical Classifier

In the absence of any labeled data in a domain, one can build sentiment-classification models that rely solely on background knowledge, such as a lexicon defining the polarity of words. Given a lexicon of positive and negative terms, one straightforward approach to using this information is to measure the frequency of occurrence of these terms in each document. The probability that a test document belongs to the positive class can then be computed as $P(+|D) = \frac{a}{a+b}$; where a and b are the number of occurrences of positive and negative terms in the document respectively. A document is then classified as positive if $P(+|D) > P(-|D)$; otherwise, the document is classified as negative. For this study, we used a lexicon of 1,267 positive and 1,701 negative terms, as labeled by human experts.

4.1.2 Pooling Multinomials

The Pooling Multinomials classifier was introduced by the authors as an approach to incorporate prior lexical knowledge into supervised learning for better text classification. In the context of sentiment analysis, such lexical knowledge is available in terms of the prior sentiment-polarity of words. Pooling Multinomials classifies unlabeled examples just as in multinomial Naïve Bayes classification (McCallum and Nigam, 1998), by predicting the class with the maximum likelihood, given by $argmax_{c_j} P(c_j) \prod_i P(w_i|c_j)$; where $P(c_j)$ is the prior probability of class c_j, and $P(w_i|c_j)$ is the probability of word w_i appearing in a snippet of class c_j. In the absence of background knowledge about the class distribution, we estimate the class priors $P(c_j)$ solely from the training data. However, unlike regular Naïve Bayes, the conditional probabilities $P(w_i|c_j)$ are computed using both the labeled examples and the lexicon of labeled features. Given two models built using labeled examples and labeled features, the multinomial parameters of such models can be aggregated through a convex combination, $P(w_i|c_j) = \alpha P_e(w_i|c_j) + (1-\alpha) P_f(w_i|c_j)$; where $P_e(w_i|c_j)$ and $P_f(w_i|c_j)$ represent the probability assigned by using the example labels and feature labels respectively, and α is the weight for combining these distributions. The weight indicates a level of confidence in each source of information, and can be computed based on the training set accuracy of the two components. The derivation and details of these models are not directly relevant to this paper, but can be found in (Melville et al., 2009).

	Q1		Q2		Q3		Q4	
	Accuracy	**AUC**	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**	**AUC**
Noise	84.62%	0.688	74.36%	0.588	74.36%	0.512	79.49%	0.441
Ambiguity	84.21%	0.715	78.95%	0.618	68.42%	0.624	84.21%	0.691
Confusion	82.50%	0.831	82.50%	0.762	80.00%	0.814	80.00%	0.645

Table 4: Effect of annotation selection on classification accuracy.

4.1.3 Results on Polarity Classification

We generated a data set of 504 snippets that had 3 or more labels for either the positive or negative class. We compare the different classification approaches using 10-fold cross-validation and present our results in Table 3. Results show that the Pooling Multinomial classifier, which makes predictions based on both the prior lexical knowledge and the training data, can learn the most from the labeled data to classify sentiments of the political blog snippets. We observe that despite the significant level of noise and ambiguity in the training data, using majority-labeled data for training still results in classifiers with reasonable accuracy.

4.2 Effect of Annotation Selection

We then evaluate the utility of the quality measures in a randomly split dataset (with 7.5% of the data in the test set). We applied each of the measures to rank the annotation examples and then divide them into 4 equal-sized training sets based on their rankings. For example, Noise-Q1 contains only the least noisy quarter of annotations and Q4 the most noisy ones.

Results in Table 4 demonstrate that the classification performance declines with the decrease of each quality measure in general, despite exceptions in the subset with the highest sentiment ambiguity (Ambiguity-Q4), the most noisy subset Q4 (Noise-Q4), and the subset yielding less overall confusion (Confusion-Q2). The results also reveal the benefits of annotation selection on efficiency: using the subset of annotations predicted in the top quality quarter achieves similar performance as using the whole training set. These preliminary results suggest that an active learning scheme which considers all three quality measures may indeed be effective in improving label quality and subsequent classification accuracy.

5 Conclusion

In this paper, we have analyzed the difference between expert and non-expert annotators in terms of annotation quality, and showed that having a single non-expert annotator is detrimental for annotating sentiment in political snippets. However, we confirmed that using multiple noisy annotations from different non-experts can still be very useful for modeling. This finding is consistent with the simulated results reported in (Sheng et al., 2008). Given the availability of many non-expert annotators on-demand, we studied three important dimensions to consider when acquiring annotations: (1) the noise level of an annotator compared to others, (2) the inherent ambiguity of an example's class label, and (3) the informativeness of an example to the current classification model. While the first measure has been studied with annotations obtained from experts (Dawid and Skene, 1979; Clemen and Reilly, 1999), the applicability of their findings on non-expert annotation selection has not been examined.

We showed how quality of labels can be improved by eliminating noisy annotators and ambiguous examples. Furthermore, we demonstrated the quality measures are useful for selecting annotations that lead to more accurate classification models. Our results suggest that a good active learning or online learning scheme in this setting should really consider all three dimensions. The way we use to integrate the different dimensions now is still preliminary. Also, our empirical findings suggest that some of the dimensions may have to be considered separately. For example, due to the divisive tendency of the most informative examples, these examples may have to be disregarded in the initial stage of annotation selection. Also, the way we use to combine these measures is still preliminary. The design and testing of such schemes are avenues for future work.

References

Giuseppe Carenini and Jackie C. K. Cheung. 2008. Extractive vs. NLG-based abstractive summarization of evaluative text: The effect of corpus controversiality. In *Proceedings of the Fifth International Natural Language Generation Conference*.

R.T. Clemen and T. Reilly. 1999. Correlations and copulas for decision and risk analysis. *Management Science*, 45:208–224.

A. P. Dawid and A. .M. Skene. 1979. Maximum liklihood estimation of observer error-rates using the em algorithm. *Applied Statistics*, 28(1):20–28.

Jeff Howe. 2008. *Crowdsourcing: Why the Power of the Crowd Is Driving the Future of Business*. Crown Business, 1 edition, August.

Michael Kaisser, Marti Hearst, and John B. Lowe. 2008. Evidence for varying search results summary lengths. In *Proceedings of ACL 2008*.

Aniket Kittur, Ed H. Chi, and Bongwon Suh. 2008. Crowdsourcing user studies with mechanical turk. In *Proceedings of CHI 2008*.

David D. Lewis and Jason Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. pages 148–156, San Francisco, CA, July.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive Bayes text classification. In *AAAI Workshop on Text Categorization*.

Prem Melville, Wojciech Gryc, and Richard Lawrence. 2009. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *KDD*.

Bo Pang and Lillian Lee. 2004. A sentiment education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of ACL 2004*.

Victor Sheng, Foster Provost, and G. Panagiotis Ipeirotis. 2008. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proceeding of KDD 2008*, pages 614–622.

Vikas Sindhwani and Prem Melville. 2008. Document-word co-regularization for semi-supervised sentiment analysis. In *Proceedings of IEEE International Conference on Data Mining (ICDM)*, pages 1025–1030, Los Alamitos, CA, USA. IEEE Computer Society.

R. Snow, B. O'Connor, D. Jurafsky, and A. Ng. 2008. Cheap and fast—but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of EMNLP 2008*.

Alexander Sorokin and David Forsyth. 2008. Utility data annotation via amazon mechanical turk. In *IEEE Workshop on Internet Vision at CVPR 08*.

Qi Su, Dmitry Pavlov, Jyh-Herng Chow, and Wendell C.Baker. 2007. Internet-scale collection of human-reviewed data. In *Proceedings of WWW 2007*.

Peter D. Turney. 2002. Thumbs up or thumbs down: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL 2002*.

Luis von Ahn and Laura Dabbish. 2004. Labeling images with a computer game. In *Proceedings of CHI 2004*, pages 319–326.

Janyce Wiebe and E. Riloff. 2005. Creating subjective and objective sentence classifiers from unannotated texts. In *Proceedings of CICLing 2005*.

Janyce Wiebe, Theresa Wilson, Rebecca Bruce, Matthew Bell, and Melanie Martin. 2004. Learning subjective language. *Computational Linguistics*, 30 (3).

Jeonghee Yi, Tetsuya Nasukawa, Razvan Bunescu, and Wayne Niblack. 2003. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing technique. In *Proceedings of the International Conference on Data Mining (ICDM)*, pages 427–434.

Hong Yu and Vasileios Hatzivassiloglou. 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of EMNLP 2003*.

Evaluating Automation Strategies in Language Documentation

Alexis Palmer, Taesun Moon, and Jason Baldridge
Department of Linguistics
The University of Texas at Austin
Austin, TX 78712
{alexispalmer,tsmoon,jbaldrid}@mail.utexas.edu

Abstract

This paper presents pilot work integrating machine labeling and active learning with human annotation of data for the language documentation task of creating interlinearized gloss text (IGT) for the Mayan language Uspanteko. The practical goal is to produce a totally annotated corpus that is as accurate as possible given limited time for manual annotation. We describe ongoing pilot studies which examine the influence of three main factors on reducing the time spent to annotate IGT: suggestions from a machine labeler, sample selection methods, and annotator expertise.

1 Introduction

Languages are dying at the rate of two each month. By the end of this century, half of the approximately 6000 extant spoken languages will cease to be transmitted effectively from one generation of speakers to the next (Crystal, 2000). Under this immense time pressure, documentary linguists seek to preserve a record of endangered languages while there are still communities of speakers to work with. Many language documentation projects target languages about which our general linguistic knowledge is nonexistent or much less than for more widely-spoken languages. The vast majority of these are individual or small-group endeavors on small budgets with little or no institutional guidance by the greater documentary linguistic community. The focus in such projects is often first on collection of data (documentation), with a following stage of linguistic analysis and description. A key part of the analysis process, detailed linguistic annotation of the recorded texts, is a time-consuming and tedious task usually occurring late in the project, if it occurs at all.

Text annotation typically involves producing interlinearized glossed text (IGT), labeling for morphology, parts-of-speech, etc., which greatly facilitates further exploration and analysis of the language. The following is IGT for the phrase *xelch li* from the Mayan language Uspanteko:[1]

(1) x- el -ch li
COM- salir -DIR DEM

Spanish: 'Salio entonces.' **English**:'Then he left.'

The levels of analysis include morpheme segmentation, transliteration of stems, and labeling of stems and morphemes with tags, some corresponding to parts-of-speech and others to semantic distinctions.

There is no single standard format for IGT. The IGT systems developed by documentation projects tend to be idiosyncratic: they may be linguistically well-motivated and intuitive, but they are unlikely to be compatible or interchangeable with systems developed by other projects. They may lack internal consistency as well. Nonetheless, IGT in a readily accessible format is an important resource that can be used fruitfully by linguists to examine hypotheses on novel data (e.g. Xia and Lewis (2007; 2008), Lewis and Xia (2008)). Furthermore, it can be used by educators and language activists to create curriculum material for mother language education and promote the survival of the language.

Despite the urgent need for such resources, IGT annotations are time consuming to create entirely by hand, and both human and financial resources are extremely limited in this domain. Thus, language

[1] KEY: COM=completive aspect, DEM=demonstrative, DIR=directional

Proceedings of the NAACL HLT Workshop on Active Learning for Natural Language Processing, pages 36–44,
Boulder, Colorado, June 2009.

documentation presents an interesting test case and an ideal context for use of machine labeling and active learning. This paper describes a series of experiments designed to assess this promise in a realistic documentation context: creation of IGT for the Mayan language Uspanteko. We systematically compare varying degrees of machine involvement in the development of IGT, from minimally involved situations where examples for tagging are selected sequentially to active learning situations where the machine learner selects samples for human tagging and suggests labels. We also discuss the challenges faced by linguists in having to learn, transcribe, analyze, and annotate a language almost simultaneously and discuss whether machine involvement reduces or compounds those challenges.

In the experiments, two documentary linguists annotate IGT for Uspanteko texts using different levels of support from a machine learned classifier. We consider the interaction of three main conditions: (1) sequential, random, or uncertainty sampling for requesting labels from an annotator, (2) suggestions or no suggestions from a machine labeler, and (3) expert versus non-expert annotator. All annotator decisions are timed, enabling the actual time cost of annotation to be measured within the context of each condition. This paper describes the Uspanteko data set we adapted for the experiments, expands on the choices described above, and reports on preliminary results from our ongoing annotation experiments.

2 Data: Uspanteko IGT

This section describes the Uspanteko corpus used for the experiments, our clean-up of the corpus, and the specific task—labeling part-of-speech and gloss tags—addressed by the experiments.

2.1 OKMA Uspanteko corpus

Our primary dataset is a corpus of texts (Pixabaj et al., 2007) in the Mayan language Uspanteko that were collected, transcribed, translated (into Spanish) and annotated as part of the OKMA language documentation project.[2] Uspanteko, a member of the K'ichee' branch of the Mayan language family, is spoken by approximately 1320 people in central Guatemala (Richards, 2003).

[2] http://www.okma.org

The corpus contains 67 texts, 32 of them glossed. Four textual genres are represented in the glossed portion of the corpus: oral histories (five texts) usually have to do with the history of the village and the community, personal experience texts (five texts) recount events from the lives of individual people in the community, and stories (twenty texts) are primarily folk stories and children's stories. The corpus also contains one recipe and one advice text in which a speaker discusses what the community should be doing to better preserve and protect the environment.

The transcriptions are based on spoken data, with attendant dysfluencies, repetitions, false starts, and incomplete sentences. Of the 284,455 words, 74,298 are segmented and glossed. This is a small dataset by computational linguistics standards but rather large for a documentation project.

2.2 Interlinearized Glossed Text

Once recordings have been made, the next tasks are typically to produce translations and transcription of the audio. Transcription is a complex and difficult process, often involving the development of an orthography for the language in parallel. The product of the transcription is raw text like the Uspanteko sample shown below (text 068, clauses 283-287):

```
  Non li in yolow rk'il kita'
tinch'ab'ex laj inyolj iin, si no ke
laj yolj jqaaj tinch'ab'ej i non qe li
xk'am rib' chuwe, non qe li lajori
non li iin yolow rk'ilaq.[3]
```

Working with the transcription, the translation, and any previously-attained knowledge about the language, the linguist next makes decisions about the division of words into morphemes and the contributions made by individual morphemes to the meaning of the word or of the sentence. IGT efficiently brings together and presents all of this information.

In the traditional four-line IGT format, morphemes appear on one line and glosses for those morphemes on the next. The gloss line includes both labels for grammatical morphemes (e.g. `PL` or `COM`) and translations of stems (e.g. *salir* or *ropa*). See the following example from Uspanteko:[4]

[3] Spanish: *Solo asi yo aprendi con él. No le hable en el idioma mio. Si no que en el idioma su papá le hablo. Y solo asi me fui acostumbrando. Solo asi ahora yo platico con ellos.*

[4] KEY: E1S=singular first person ergative, INC=incompletive, PART=particle, PREP=preposition, PRON=pronoun, NEG=negation,

(2) `Kita' tinch'ab'ej laj inyolj iin`

(3)
kita'	t-in-ch'abe-j	laj	in-yolj	iin	
NEG	INC-E1S-hablar-SC	PREP	E1S-idioma	yo	
PART	TAM-PERS-VT-SUF	PREP	PERS-S	PRON	

'No le hablo en mi idioma.'
('I don't speak to him in my language.')

Most commonly, IGT is presented in a four-tier format. The first tier (2) is the raw, unannotated text. The second (first line of (3)) is the same text with each word morphologically segmented. The third tier (second line of (3)), the gloss line, is a combination of Spanish translations of the Uspanteko stems and gloss tags representing the grammatical information encoded by affixes and stand-alone morphemes. The fourth tier (fourth line of (3)) is a translation in the target language of documentation.

Some interlinear texts include other project-defined tiers. OKMA uses a fifth tier (third line of (3)), described as the word-class line. This line is a mix of traditional POS tags, positional labels (e.g. suffix, prefix), and broader linguistic categories like `TAM` for tense-aspect-mood.

2.3 Cleaning up the OKMA annotations

The OKMA annotations were created using Shoebox,[5] a standard tool used by documentary linguists for lexicon management and IGT creation. To develop a corpus suitable for these studies, it was necessary to put considerable effort into normalizing the original OKMA source annotations. Varied levels of linguistic training of the original annotators led to many inconsistencies in the original annotations. Also, Shoebox (first developed in 1987) uses a custom, pre-XML whitespace delimited data format, making normalization especially challenging. Finally, not all of the texts are fully annotated. Almost half of the 67 texts are just transcriptions, several texts are translated but not further analyzed, and several others are only partially annotated at text level, clause level, word level, or morpheme level. It was thus necessary to identify complete texts for use in our experiments. Some missing labels in nearly-complete texts were filled in by the expert annotator.

A challenge for representing IGT in a machine-readable format is maintaining the links between the source text morphemes in the second tier and the morpheme-by-morpheme glosses in the third tier. The standard Shoebox output format, for example, enforces these links through management of the number of spaces between items in the output. To address this, we converted the cleaned annotations into IGT-XML (Palmer and Erk, 2007) with help from the Shoebox/Toolbox interfaces provided in the Natural Language Toolkit (Robinson et al., 2007). Automating the transformation from Shoebox format to IGT-XML's hierarchical format required cleaning up tier-to-tier alignment and checking segmentation in some cases where morphemes and glosses were misaligned, as in (5) below.[6]

(4) `Non li in yolow rk'il`

(5)
Non	li	in	yolow	r-k'il	
DEM	DEM	yo	platicar	AP	E3s.-SR
DEM	DEM	PRON	VI	SUF	PERS SREL

'Solo asi yo aprendi con él.'

Here, the number of elements in the morpheme tier (first line of (5)) does not match the number of elements in the gloss tier (second line of (5)). The problem is a misanalysis of `yolow`: it should be segmented `yol-ow` with the gloss `platicar-AP`. Automating this transformation has the advantage of identifying such inconsistencies and errors.

There also were many low-level issues that had to be handled, such as checking and enforcing consistency of tags. For example, the tag `E3s.` in the gloss tier of (5) is a typo; the correct tag is `E3S`. The annotation tool used in these studies does not allow such inconsistencies to occur.

2.4 Target labels

There are two main tasks in producing IGT: word segmentation (determination of stems and affixes) and glossing each segment. Stems and affixes each get a different type of gloss: the gloss of a stem is typically its translation whereas the gloss of an affix is a label indicating its grammatical role. The additional word-class line provides part-of-speech information for the stems, such as VT for *salir*.

Complete prediction of segmentation, gloss translations and labels is our ultimate goal for aiding IGT

S=sustantivo (noun), SC=category suffix, SUF=suffix, TAM=tense/aspect/mood, VT=transitive verb

[5] `http://www.sil.org/computing/shoebox/`

[6] KEY: AP=antipassive, DEM=demonstrative, E3S=singular third person ergative, PERS=person marking, SR/SREL=relational noun, VI=intransitive verb

creation with automation. Here, we study the potential for improving annotation efficiency for the more limited task of predicting the gloss label for each affix and the part-of-speech label for each stem. Thus, the experiments aim to produce a single label for each morpheme. We assume that words have been pre-segmented and we ignore the gloss translations.

The target representation in these studies is an additional tier which combines gloss labels for affixes and stand-alone morphemes with part-of-speech labels for stems. Example (6) repeats the clause in (4), adding this new combined tier. Stem labels are given in bold text, and affix labels in plain text.

(6) `Non li in yolow rk'il`

(7) `Non li in yol-ow r-k'il`
DEM DEM **PRON** **VI**-AP E3S-SR

'Solo asi yo aprendi con él.'

A simple procedure was used to create the new tier. For each morpheme, if a gloss label (such as `DEM` or `E3S`) appears on the gloss line (second line of (3)), we select that label. If what appears is a stem translation, we instead select the part-of-speech label from the next tier down (third line of (3)).

In the entire corpus, sixty-nine different labels appear in this combined tier. The following table shows the five most common part-of-speech labels (left) and the five most common gloss labels (right). The most common label, `S`, accounts for 11.3% of the tokens in the corpus.

S	noun	7167	**E3S**	sg.3p. ergative	3433
ADV	adverb	6646	**INC**	incompletive	2835
VT	trans. verb	5122	**COM**	completive	2586
VI	intrans. verb	3638	**PL**	plural	1905
PART	particle	3443	**SREL**	relational noun	1881

3 Integrated annotation and automation

The experimental framework described in this section is designed to model and evaluate real-time integration of human annotation, active learning strategies, and output from machine-learned classifiers. The task is annotation of morpheme-segmented texts from a language documentation project (sec. 2).

3.1 Tools and resources

Integrating automated support and human annotation in this context requires careful coordination of three components: 1) presenting examples to the annotator and storing the annotations, 2) training and evaluation of tagging models using data labeled by the annotator, and 3) selecting new examples for annotation. The processes are managed and coordinated using the OpenNLP IGT Editor.[7] The annotation component of the tool, and in particular the user interface, is built on the Interlinear Text Editor (Lowe et al., 2004).

For tagging we use a strong but simple standard classifier. There certainly are many other modeling strategies that could be used, for example a conditional random field (as in Settles and Craven (2008)), or a model that deals differently with POS labels and morpheme gloss labels. Nonetheless, a documentary linguistics project would be most likely to use a straightforward, off-the-shelf labeler, and our focus is on exploring different annotation approaches in a realistic documentation setting rather than building an optimal classifier. To that end, we use a standard maximum entropy classifier which predicts the label for a morpheme based on the morpheme itself plus a window of two morphemes before and after. Standard features used in part-of-speech taggers are extracted from the morpheme to help with predicting labels for previously unseen stems and morphemes.

3.2 Annotators and annotation procedures

A practical goal of these studies is to explore best practices for using automated support to create fully-annotated texts of the highest quality possible within fixed resource limits. For producing IGT, one of the most valuable resources is the time of a linguist with language-specific expertise. Documentary projects may also (or instead) have access to a trained linguist without prior experience in the language. We compare results from two annotators with different levels of exposure to the language. Both are trained linguists who specialize in language documentation and have extensive field experience.[8]

The first, henceforth referred to as the **expert annotator**, has worked extensively on Uspanteko, including writing a grammar of the language and

[7] `http://igt.sourceforge.net/`

[8] It should be noted that these are pilot studies. With just two annotators, the annotation comparisons are suggestive but not conclusive. Even so, this scenario accurately reflects the resource limitations encountered in documentation projects.

contributing to the publication of an Uspanteko-Spanish dictionary (Ángel Vicente Méndez, 2007). She is a native speaker of K'ichee', a closely-related Mayan language. The second annotator, the **non-expert annotator**, is a doctoral student in language documentation with no prior experience with Uspanteko and only limited previous knowledge of Mayan languages. Throughout the annotation process, the non-expert annotator relied heavily on the Uspanteko-Spanish dictionary. Both annotators are fluent speakers of Spanish, the target translation and glossing language for the OKMA texts.

In many annotation projects, labeling of training data is done with reference to a detailed annotation manual. In the language documentation context, a more usual situation is for the annotator(s) to work from a set of agreed-upon conventions but without strict annotation guidelines. This is not because documentary linguists lack motivation or discipline but simply because many aspects of the language are unknown and the analysis is constantly changing.

In the absence of explicit written annotation guidelines, we use an annotation training process for the annotators to learn the OKMA annotation conventions. Two seed sets of ten clauses each were selected to be used both for human annotation training and for initial classifier training. The first ten clauses of the first text in the training data were used to seed model training for the sequential selection cases (see 3.4). The second set of ten were randomly selected from the entire corpus and used to seed model training for both random and uncertainty sampling.

These twenty clauses were used to provide initial guidance to the annotators. With the aid of a list of possible labels and the grammatical categories they correspond to, each annotator was asked to label the seed clauses, and these labels were compared to the gold standard labels. Annotators were told which labels were correct and which were incorrect, and the process was repeated until all morphemes were correctly labeled. In some cases during this training phase, the correct label for a morpheme was supplied to the annotator after several incorrect guesses.

3.3 Suggesting labels

We consider two situations with respect to the contribution of the classifier: a **suggest** condition in which the labels predicted by the machine learner are shown to the annotator as she begins labeling a selected clause, and a **no-suggest** condition in which the annotator does not see the predicted labels.

In the suggest cases, the annotator is shown the label assigned the greatest likelihood by the tagger as well as a list of several highly-likely labels, ranked according to likelihood. To be included on this list, a label must be assigned a probability greater than half that of the most-likely label. In the no-suggest cases, the annotator has access to a list of the labels previously seen in the training data for a given morpheme, ranked in order of frequency of occurrence with the morpheme in question; this is similar to the input an annotator gets while glossing texts in Shoebox/Toolbox. Specifically, Shoebox/Toolbox presents previously seen glosses and labels for a given morpheme in alphabetic order.

3.4 Sample selection

We consider three methods of selecting examples for annotation–sequential (**seq**), random (**rand**), and uncertainty sampling (**al**)–and the performance of each method in both the **suggest** and the **no-suggest** setups. For uncertainty sampling, we measure uncertainty of a clause as the average entropy per morpheme (i.e., per labeling decision).

3.5 Measuring annotation cost

Not all examples take the same amount of effort to annotate. Even so, the bulk of the literature on active learning assumes some sort of unit cost to determine the effectiveness of different sample selection strategies. Examples of unit cost measurements include the number of documents in text classification, the number of sentences in part-of-speech tagging (Settles and Craven, 2008), or the number of constituents in parsing (Hwa, 2000). These measures are convenient for performing active learning simulations, but awareness has grown that they are not truly representative measures of the actual cost of annotation (Haertel et al., 2008a; Settles et al., 2008), with Ngai and Yarowsky (2000) being an early exception to the unit-cost approach. Also, Baldridge and Osborne (2004) use discriminants in parse selection, which are annotation decisions that they later showed correlate with timing information (Baldridge and Osborne, 2008).

The cost of annotation ultimately comes down to

money. Since annotator pay may be variable but will (under standard assumptions) be constant for a given annotator, the best approximation of likely cost savings is to measure the time taken to annotate under different levels of automated support. This is especially important in sample selection and its interaction with automated suggestions: active learning seeks to find more informative examples, and these will most likely involve more difficult decisions, decreasing annotation quality and/or increasing annotation time (Hachey et al., 2005). Thus, we measure cost in terms of the time taken by each annotator on each example. This allows us to measure the actual time taken to produce a given labeled data set, and thus compare the effectiveness of different levels of automated support plus their interaction with annotators of different levels of expertise.

Recent work shows that paying attention to predicted annotation cost in sample selection itself can increase the effectiveness of active learning (Settles et al., 2008; Haertel et al., 2008b). Though we have not explored cost-sensitive selection here, the scenario described here is an appropriate test ground for it: in fact, the results of our experiments, reported in the next section, provide strong evidence for a real natural language annotation task that active learning selection with cost-sensitivity is indeed sub-optimal.

4 Discussion

This section presents and discusses preliminary results from the ongoing annotation experiments. The Uspanteko corpus was split into training, development, and held-out test sets, roughly 50%, 25%, and 25%. Specifically, the training set of 21 texts contains 38802 words, the development set of 5 texts contains 16792 words, and the held-out test set, 6 texts, contains 18704 words. These are small datasets, but the size is realistic for computational work on endangered languages.

When measuring the performance of annotators, factors like fatigue, frustration, and especially the annotator's learning process must be considered. Annotators improve as they see more examples (especially the non-expert annotator). To minimize the impact of the annotator's learning process on the results, annotation is done in rounds. Each round consists of ten clauses from each of the six experimental

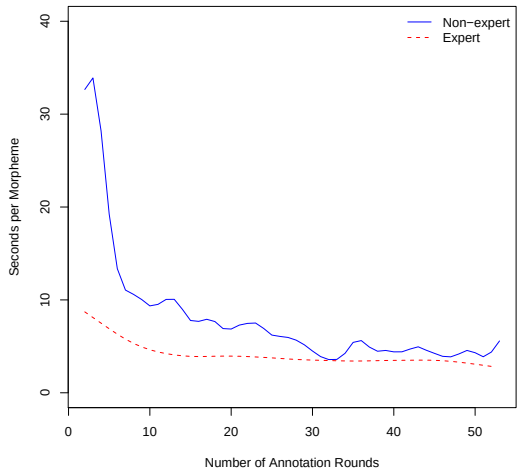


Figure 1: Average annotation time (in seconds per morpheme) over annotation rounds, averaged over all six conditions for each annotator.

cases for each annotator. The newly-labeled clauses are then added to the labeled training data, and a new tagging model is trained on the updated training set and evaluated on the development set. Both annotators have completed fifty-one rounds of annotation so far, labeling 510 clauses for each of the six experimental conditions. The average number of morphemes labeled is 3059 per case. Because the annotation experiments are ongoing, we discuss results in terms of the trends seen thus far.

4.1 Annotator speed

The expert annotator showed a small increase in speed after an initial familiarization period, and the non-expert showed a dramatic increase. Figure 1 plots the number of seconds taken per morpheme over the course of annotation, averaged over all six conditions for each annotator. The slowest, fastest, and mean rates, in seconds per morpheme, for the expert annotator were 12.60, 1.89, and 4.14, respectively. For the non-expert, they were 59.71, 1.90, and 8.03.

4.2 Accuracy of model on held-out data

Table 1 provides several measures of the current state of annotation in all 12 conditions after 51 rounds of annotation. The sixth column, labeled

Anno	Suggest	Select	Time (sec)	#Morphs	Model Accuracy	Total Accuracy of Annotation
NonExp	N	Seq	23739.79	3314	63.28	63.92
NonExp	N	Rand	22721.11	2911	**68.36**	68.69
NonExp	N	AL	23755.71	2911	68.26	67.84
NonExp	Y	Seq	21514.05	2887	66.55	66.89
NonExp	Y	Rand	22189.68	3002	**68.41**	68.73
NonExp	Y	AL	25731.57	2750	67.63	67.30
Exp	N	Seq	11862.39	3354	61.15	61.88
Exp	N	Rand	11665.10	3043	64.60	64.91
Exp	N	AL	13894.14	3379	**66.74**	66.47
Exp	Y	Seq	11758.74	2892	61.12	61.48
Exp	Y	Rand	11426.85	2979	60.13	60.57
Exp	Y	AL	16253.40	3296	**63.30**	63.15

Table 1: After 51 rounds of annotation: ModelAcc=accuracy on development set, TotalAnnoAcc=accuracy of fully-labeled corpus

ModelAcc, shows the accuracy of models on the development data. This represents a unit cost assumption at the clause level: measured this way, the results would suggest that the non-expert was best served by random selection, with no effect from machine suggestions. For the expert, they suggest active learning without suggestions is best, and that suggestions actually hurt effectiveness.

4.3 Accuracy of fully-labeled corpus

We are particularly concerned with the question of how to develop a fully-labeled corpus with the highest level of accuracy, given a finite set of resources. Thus, we combine the portion of the training set labeled by the human annotator with the results of tagging the *remainder* of the training set with the model trained on those annotations. The rightmost column of Table 1, labeled **Total Accuracy of Annotation**, shows the accuracy of the fully labeled training set (part human, part machine labels) after 51 rounds. These accuracies parallel the model accuracies: random selection is best for the non-expert annotator, and uncertainty selection is best for the expert.

Since this tagging task involves labeling morphemes, a clause cost assumption is not ideal—e.g., active learning tends to select longer clauses and thereby obtains more labels. To reflect this, a subclause cost can help: here we use the number of morphemes annotated. The column labeled **Tokens** in Table 2 shows the total accuracy achieved in each condition when human annotation ceases at 2750 morphemes. The figure in parentheses is the cumulative annotation time at the morpheme cut-off point. Here, the non-expert does best: he took great care with the annotations and was clearly not tempted to accept erroneous suggestions from the machine labeler. In contrast, the expert does seem to have accepted many bad machine suggestions.

Anno	Suggest	Select	Time (11427 sec)	Tokens (time) (2750 morphs)
NonExp	N	Seq	55.01	59.80 (21678 secs)
NonExp	N	Rand	59.95	**68.68** (22069 secs)
NonExp	N	AL	59.86	67.70 (22879 secs)
NonExp	Y	Seq	60.27	66.79 (21053 secs)
NonExp	Y	Rand	62.96	68.38 (21194 secs)
NonExp	Y	AL	59.18	67.30 (25732 secs)
Exp	N	Seq	61.21	59.18 (10110 secs)
Exp	N	Rand	64.92	64.42 (10683 secs)
Exp	N	AL	**65.72**	65.74 (11826 secs)
Exp	Y	Seq	61.47	61.47 (11436 secs)
Exp	Y	Rand	60.57	61.16 (10934 secs)
Exp	Y	AL	61.54	62.87 (13957 secs)

Table 2: For given cost, accuracy of fully-labeled corpus.

Morpheme unit cost is more fine-grained than clause-level cost, but it hides the fact that the expert annotator needed far less time to produce a corpus of higher overall labeled quality than the non-expert. This can be seen in the **Time** column of Table 2, which gives the total annotation accuracy when 11427 seconds are alloted for human labeling. The expert annotator achieved the highest accuracy for total labeling of the training set using active learning without machine label suggestions. Active learning helps the non-expert as well, but his best condition is random selection with machine labels.

4.4 Annotator accuracy by round

Active learning clearly selects harder examples that hurt the non-expert's performance. To see this clearly, we measured the accuracy of the annotators' labels for each round of each experimental setup,

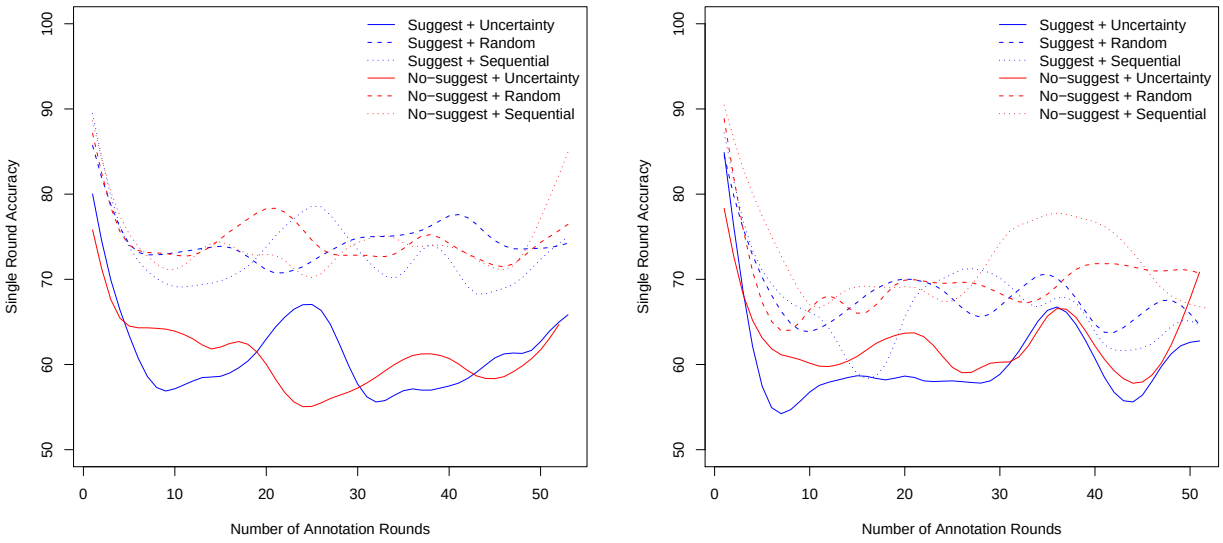


Figure 2: Single round accuracy per round for each experiment type by: (a) non-expert annotator, (b) expert annotator

given in Fig. 2. It is not clear at this stage whether the tag suggestions by the machine labeler are helpful to human annotation. It is useful to compare the cases where the machine learner is not involved in example selection (i.e. random and sequential) to uncertainty sampling, which does involve the machine learner. One thing that is apparent is that when active learning is used to select samples for annotation, both the expert and non-expert annotator have a harder time providing correct tags. A point of contrast between the expert and non-expert is that the non-expert generally outperforms the expert on label accuracy in the non-active learning scenarios. The non-expert was very careful with his labeling decisions, but also much slower than the expert. In the end, speedier annotation rates allowed the expert annotator to achieve higher accuracies in less time.

5 Conclusion

We have described a set of ongoing pilot experiments designed to test the utility of machine labeling and active learning in the context of documentary linguistics. The production of IGT is a realistic annotation scenario which desperately needs labeling efficiency improvements. Our preliminary results suggest that both machine labeling and active learning can increase the effectiveness of annotators, but they interact quite strongly with the expertise of the annotators. In particular, though active learning works well with the expert annotator, for a non-expert annotator it seems that random selection is a better choice. However, we stress that our annotation experiments are ongoing. Active learning is often less effective early in the learning curve, especially when automated label suggestions are provided, because the model is not yet accurate enough to select truly useful examples, nor to suggest labels for them reliably (Baldridge and Osborne, 2004). Thus, we expect automation via uncertainty sampling and/or suggestion may gather momentum and outpace random selection and/or no suggestions by wider margins as annotation continues.

Acknowledgments

This work is funded by NSF grant BCS 06651988 “Reducing Annotation Effort in the Documentation of Languages using Machine Learning and Active Learning.” Thanks to Katrin Erk, Nora England, Michel Jacobson, and Tony Woodbury; and to annotators Telma Kaan Pixabaj and Eric Campbell. Finally, thanks to the anonymous reviewers for valuable feedback.

References

Miguel Ángel Vicente Méndez. 2007. *Diccionario bilingüe Uspanteko-Español. Cholaj Tzijb'al li Uspanteko*. Okma y Cholsamaj, Guatemala.

Jason Baldridge and Miles Osborne. 2004. Active learning and the total cost of annotation. In *Proceedings of Empirical Approaches to Natural Language Processing (EMNLP)*.

Jason Baldridge and Miles Osborne. 2008. Active learning and logarithmic opinion pools for HPSG parse selection. *Natural Language Engineering*, 14(2):199–222.

David Crystal. 2000. *Language Death*. Cambridge University Press, Cambridge.

Ben Hachey, Beatrice Alex, and Markus Becker. 2005. Investigating the effects of selective sampling on the annotation task. In *Proceedings of the 9th Conference on Computational Natural Language Learning*, Ann Arbor, MI.

Robbie Haertel, Eric Ringger, Kevin Seppi, James Carroll, and McClanahan Peter. 2008a. Assessing the costs of sampling methods in active learning for annotation. In *Proceedings of ACL-08: HLT, Short Papers*, pages 65–68, Columbus, Ohio, June. Association for Computational Linguistics.

Robbie A. Haertel, Kevin D. Seppi, Eric K. Ringger, and James L. Carroll. 2008b. Return on investment for active learning. In *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*. ACL Press.

Rebecca Hwa. 2000. Sample selection for statistical grammar induction. In *Proceedings of the 2000 Joint SIGDAT Conference on EMNLP and VLC*, pages 45–52, Hong Kong, China, October.

William Lewis and Fei Xia. 2008. Automatically identifying computationally relevant typological features. In *Proceedings of IJCNLP-2008*, Hyderabad, India.

John Lowe, Michel Jacobson, and Boyd Michailovsky. 2004. Interlinear text editor demonstration and projet archivage progress report. In *4th EMELD workshop on Linguistic Databases and Best Practice*, Detroit, MI.

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 of the Association for Computational Linguistics*, pages 117–125, Hong Kong.

Alexis Palmer and Katrin Erk. 2007. IGT-XML: An XML format for interlinearized glossed text. In *Proceedings of the Linguistic Annotation Workshop (LAW-07), ACL07*, Prague.

Telma Can Pixabaj, Miguel Angel Vicente Méndez, María Vicente Méndez, and Oswaldo Ajcot Damián. 2007. Text collections in Four Mayan Languages. Archived in The Archive of the Indigenous Languages of Latin America.

Michael Richards. 2003. *Atlas lingüístico de Guatemala*. Servipresna, S.A., Guatemala.

Stuart Robinson, Greg Aumann, and Steven Bird. 2007. Managing fieldwork data with Toolbox and the Natural Language Toolki t. *Language Documentation and Conservation*, 1:44–57.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1070–1079, Honolulu, Hawaii, October. Association for Computational Linguistics.

Burr Settles, Mark Craven, and Lewis Friedland. 2008. Active learning with real annotation costs. In *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, pages 1069–1078. ACL Press.

Fei Xia and William Lewis. 2007. Multilingual structural projection across interlinear text. In *Proceedings of HLT/NAACL 2007*, Rochester, NY.

Fei Xia and William Lewis. 2008. Repurposing theoretical linguistic data for tool development antd search. In *Proceedings of IJCNLP-2008*, Hyderabad, India.

A Web Survey on the Use of Active Learning to Support Annotation of Text Data

Katrin Tomanek
Jena University Language & Information Engineering Lab
Friedrich-Schiller-Universität Jena
Fürstengraben 30, D-07743 Jena, Germany
katrin.tomanek@uni-jena.de

Fredrik Olsson
SICS
Box 1263
SE-164 29 Kista, Sweden
fredrik.olsson@sics.se

Abstract

As supervised machine learning methods for addressing tasks in natural language processing (NLP) prove increasingly viable, the focus of attention is naturally shifted towards the creation of training data. The manual annotation of corpora is a tedious and time consuming process. To obtain high-quality annotated data constitutes a bottleneck in machine learning for NLP today. Active learning is one way of easing the burden of annotation. This paper presents a first probe into the NLP research community concerning the nature of the annotation projects undertaken in general, and the use of active learning as annotation support in particular.

1 Introduction

Supervised machine learning methods have been successfully applied to many NLP tasks in the last few decades. While these techniques have shown to work well, they require large amounts of labeled training data in order to achieve high performance. Creating such training data is a tedious, time consuming and error prone process. Active learning (AL) is a supervised learning technique that can be used to reduce the annotation effort. The main idea in AL is to put the machine learner in control of the data from which it learns; the learner can ask an oracle (typically a human) about the labels of the examples for which the model learned so far makes unreliable predictions. The active learning process takes as input a set of labeled examples, as well as a larger set of unlabeled examples, and produces a classifier and a relatively small set of newly labeled data. The overall goal is to create as good a classifier as possible, without having to mark-up and supply the learner with more data than necessary. AL aims at keeping the human annotation effort to a minimum, only asking the oracle for advice where the training utility of the result of such a query is high. Settles (2009) gives a detailed overview of the literature on AL.

It has been experimentally shown that AL can indeed be successfully applied to a range of NLP tasks including, e.g., text categorization (Lewis and Gale, 1994), part-of-speech tagging (Dagan and Engelson, 1995; Ringger et al., 2007), parsing (Becker and Osborne, 2005), and named entity recognition (Shen et al., 2004; Tomanek et al., 2007). Despite that somewhat impressive results in terms of reduced annotation effort have been achieved by such studies, it seems that AL is rarely applied in real-life annotation endeavors.

This paper presents the results from a web survey we arranged to analyze the extent to which AL has been used to support the annotation of textual data in the context of NLP, as well as addressing the reasons to why or why not AL has been found applicable to a specific task. Section 2 describes the survey in general, Section 3 introduces the questions and presents the answers received. Finally, the answers received are discussed in Section 4.

2 The Survey

The survey was realized in the form of a web-based questionnaire; the primary reason for this approach, as opposed to reading and compiling information

Proceedings of the NAACL HLT Workshop on Active Learning for Natural Language Processing, pages 45–48,
Boulder, Colorado, June 2009.

from academic publications, was that we wanted to free ourselves and the participants from the dos and don'ts common to the discourse of scientific papers.

The survey targeted participants who were involved in the annotation of textual data intended for machine learning for all kinds of NLP tasks. It was announced on the following mailing lists: BioNLP, Corpora, UAI List, ML-news, SIG-IRlist, Linguist list, as well as lists reaching members of SIGANN, SIGNLL, and ELRA. By utilizing these mailing lists, we expect to have reached a fairly large portion of the researchers likely to participate in annotation projects for NLP. The questionnaire was open February 6–23, 2009.

After an introductory description and one initial question, the questionnaire was divided into two branches. The first branch was answered by those who had used AL to support their annotation, while the second branch was answered by those who had not. Both branches shared a common first part about the general set-up of the annotation project under scrutiny. The second part of the AL-branch focused on experiences made with applied AL. The second part of the non AL-branch asked questions about the reasons why AL had not been used. Finally, the questionnaire was concluded by a series of questions targeting the background of the participant.

The complete survey can be downloaded from `http://www.julielab.de/ALSurvey`.

3 Questions and answers

147 people participated in the survey. 54 completed the survey while 93 did not, thus the overall completion rate was 37 %. Most of the people who did not complete the questionnaire answered the first couple of questions but did not continue. Their answers are not part of the discussion below. We refrain from a statistically analysis of the data but rather report on the distribution of the answers received.

Of the people that finished the survey, the majority (85 %) came from academia, with the rest uniformly split between governmental organizations and industry. The educational background of the participants were mainly computational linguistics (46 %), general linguistics (22 %), and computer science (22 %).

3.1 Questions common to both branches

Both the AL and the non-AL branch were asked several questions about the set-up of the annotation project under scrutiny. The questions concerned, e.g., whether AL had been used to support the annotation process, the NLP tasks addressed, the size of the project, the constitution of the corpus annotated, and how the decision when to stop the annotation process was made.

The use of AL as annotation support. The first question posed was whether people had used AL as support in their annotation projects. 11 participants (20 %) answered this question positively, while 43 (80 %) said that they had not used AL.

The task addressed. Most AL-based annotation projects concerned the tasks information extraction (IE) (52 %), document classification (17.6 %), and (word sense) disambiguation (17.6 %). Also in non AL-based projects, most participants had focused on IE tasks (36.8 %). Here, syntactic tasks including part-of-speech tagging, shallow, and deep parsing were also often considered (19.7 %). Textual phenomena, such as coreferences and discourse structure (9.6 %), and word sense disambiguation (5.5 %) formed two other answer groups. Overall, the non AL-based annotation projects covered a wider variety of NLP tasks than the AL-based ones. All AL-based annotation projects concerned English texts, whereas of the non-AL projects only 62.8 % did.

The size of the project. The participants were also asked for the size of the annotation project in terms of number of units annotated, number of annotators involved and person months per annotator. The average number of person months spent on non AL-projects was 21.2 and 8.7 for AL-projects. However, these numbers are subject to a high variance.

The constitution of the corpus. Further, the participants were asked how the corpus of unlabeled instances was selected.[1] The answer options included *(a)* taking all available instances, *(b)* a random subset of them, *(c)* a subset based on keywords/introspection, and *(d)* others. In the AL-branch, the answers were uniformly distributed be-

[1] The unlabeled instances are used as a pool in AL, and as a corpus in non AL-based annotation.

tween the alternatives. In the non AL-branch, the majority of participants had used alternatives *(a)* (39.5 %) and *(b)* (34.9 %).

The decision to stop the annotation process. A last question regarding general annotation project execution concerned the stopping of the annotation process. In AL-based projects, evaluation on a held-out gold standard (36.5 %) and the exhaustion of money or time (36.5 %) were the major stopping criteria. Specific stopping criteria based on AL-internal aspects were used only once, while in two cases the annotation was stopped because the expected gains in model performance fell below a given threshold.

In almost half (47.7 %) of the non AL-based projects the annotation was stopped since the available money or time had been used up. Another major stopping criterion was the fact that the complete corpus was annotated (36 %). Only in two cases annotation was stopped based on an evaluation of the model achievable from the corpus.

3.2 Questions specific to the AL-branch

The AL-specific branch of the questionnaire was concerned with two aspects: the learning algorithms involved, and the experiences of the participants regarding the use of AL as annotation support. Percentages presented below are all related to the 11 persons who answered this branch.

Learning algorithms used. As for the AL methods applied, there was no single most preferred approach. 27.3 % had used uncertainty sampling, 18.2 % query-by-committee, another 18.2% error reduction-based approaches, and 36.4 % had used an "uncanonical" or totally different approach which was not covered by any of these categories. As base learners, maximum-entropy based approaches as well as Support-Vector machines were most frequently used (36.4 % each).

Experiences. When asked about their experiences, the participants reported that their expectations with respect to AL had been partially (54.4 %) or fully (36.3 %) met, while one of the participants was disappointed. The AL participants did not leave many experience reports in the free text field. From the few received, it was evident that the sampling complexity and the resulting delay or idle time of the annotators, as well as the interface design are critical issues in the practical realization of AL as annotation support.

3.3 Question specific to the non-AL branch

The non AL-specific branch of the questionnaire was basically concerned with why people did not use AL as annotation support and whether this situation could be changed. The percentages given below are related to the 43 people who answered this particular part of the questionnaire.

Why was not AL used? Participants could give multiple answers to this question. Many participants had either never heard of AL (11 %) or did not use AL due to insufficient knowledge or expertise (26 %). The implementational overhead to develop an AL-enabled annotation editor kept 17.8 % of the participants from using AL. Another 19.2 % of the participants stated that their project specific requirements did not allow them to use AL. Given the comments given in the free text field, it can be deduced that this was often the case when people wanted to create a corpus that could be used for a multitude of purposes (such as building statistics on, cross-validation, learning about the annotation task per se, and so forth) and not just for classifier training. In such scenarios, the sampling bias introduced by AL is certainly disadvantageous. Finally, about 20.5 % of the participants were not convinced that AL would work well in their scenario or really reduce annotation effort. Some participants stated in their free form comments that while they believed AL would reduce the amount of instances to be annotated it would probably not reduce the overall annotation time.

Would you consider using AL in future projects? According to the answers of another question of the survey, 40 % would in general use AL, while 56 % were sceptical but stated that they would possibly use a technique such as AL.

4 Discussion

Although it cannot be claimed that the data collected in this survey is representative for the NLP research community as a whole, and the number of participants was too low to draw statistically firm conclusions, some interesting trends have indeed been

discovered within the data itself. The conclusions drawn in this section are related to the answers provided in light of the questions posed in the survey.

The questionnaire was open to the public and was not explicitly controlled with respect to the distribution of characteristics of the sample of the community that partook in it. One effect of this, coupled with the fact that the questionnaire was biased towards those familiar with AL, is that we believe that the group of people that have used AL are overrepresented in the data at hand. However, this cannot be verified. Nevertheless, given this and the potential reach of the mailing lists used for announcing the survey, it is remarkable that not more than 20 % (11 out of 54) of the participants had used AL as annotation support.

The doubts of the participants who did not use AL towards considering the technique as a potential aid in annotation in essence boil down to the absence of an AL-based annotation editor, as well as the difficulty in estimating the effective reduction in effort (such as time, money, labor) that the use of AL imply. Put simply: Can AL for NLP really cut annotation costs? Can AL for NLP be practically realized without too much overhead in terms of implementation and education of the annotator? Research addressing the former question is ongoing which is shown, e.g., by the recent Workshop on Cost-Sensitive Learning held in conjunction with the Neural Information Processing Systems Conference 2008. As for the latter question, there is evidently a need of a general framework for AL in which (specialized) annotation editors can be realized. Also, hand-in-hand with the theoretical aspects of AL and their practical realizations in terms of available software packages, there clearly is a need for usage and user studies concerning the effort required by human annotators operating under AL-based data selection schemes in real annotation tasks.

Two things worth noticing among the answers from participants of the survey that had used AL include that most of these participants had positive experiences from using AL, although turn-around time and consequently the idle time of the annotator remains a critical issue; and that English was the only language addressed. This is somewhat surprising given that AL seems to be a technique well suited for bootstrapping language resources for, e.g., so called "under resourced" languages. Also we were surprised by the fact that both in AL and non-AL projects rather "unsophisticated" criteria were used to decide about the stopping of annotation projects.

Acknowledgements

The first author was funded by the German Ministry of Education and Research within the STEM-NET project (01DS001A-C) and the EC within the BOOTStrep project (FP6-028099). We wish to thank Björn Gambäck for commenting and proofreading drafts of this paper.

References

Markus Becker and Miles Osborne. 2005. A two-stage method for active learning of statistical grammars. In *Proc. of the 19th International Joint Conference on Artificial Intelligence*, pages 991–996.

Ido Dagan and Sean P. Engelson. 1995. Committee-based sampling for training probabilistic classifiers. In *Proc. of the 12th International Conference on Machine Learning*, pages 150–157.

David D. Lewis and William A. Gale. 1994. A Sequential Algorithm for Training Text Classifiers. In *Proc. of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12.

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 *Proc. of the Linguistic Annotation Workshop*, pages 101–108.

Burr Settles. 2009. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin-Madison.

Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew-Lim Tan. 2004. Multi-criteria-based active learning for named entity recognition. In *Proc. of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 589–596.

Katrin Tomanek, Joachim Wermter, and Udo Hahn. 2007. An approach to text corpus construction which cuts annotation costs and maintains reusability of annotated data. In *Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 486–495.

Active Dual Supervision: Reducing the Cost of Annotating Examples and Features

Prem Melville
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598
pmelvil@us.ibm.com

Vikas Sindhwani
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598
vsindhw@us.ibm.com

Abstract

When faced with the task of building machine learning or NLP models, it is often worthwhile to turn to active learning to obtain human annotations at minimal costs. Traditional active learning schemes query a human for labels of intelligently chosen examples. However, human effort can also be expended in collecting alternative forms of annotations. For example, one may attempt to learn a text classifier by labeling class-indicating words, instead of, or in addition to, documents. Learning from two different kinds of supervision brings a new, unexplored dimension to the problem of active learning. In this paper, we demonstrate the value of such active dual supervision in the context of sentiment analysis. We show how interleaving queries for both documents and words significantly reduces human effort – more than what is possible through traditional one-dimensional active learning, or by passive combinations of supervisory inputs.

1 Introduction

As a canonical running example for the theme of this paper, consider the problem of *sentiment analysis* (Pang and Lee, 2008). Given a piece of text as input, the desired output is a *polarity score* that indicates whether this text expresses a positive or negative opinion towards a topic of interest. From a machine learning viewpoint, this problem may be posed as a typical binary text classification task. Sentiment, however, is often conveyed with subtle linguistic mechanisms such as sarcasm, negation and the use of highly domain-specific and contextual cues. This brings a multi-disciplinary flavor to the problem, drawing interest from both Natural Language Processing and Machine Learning communities.

Many methodologies proposed in these disciplines share a common limitation that their performance is bounded by the amount and quality of labeled data. However, they differ conceptually in the *type* of human effort they require. On one hand, supervised machine learning techniques require human effort in acquiring *labeled examples*, which requires reading documents and annotating them with their aggregate sentiment. On the other hand, dictionary-based NLP systems require human effort in collecting *labeled features*: for example, in the domain of movie reviews, words that evoke positive sentiment (e.g., "mesmerizing", "thrilling" etc) may be labeled positive, while words that evoke negative sentiment (e.g., "boring","disappointing") may be labeled negative. This kind of annotation requires a human to condense prior linguistic experience with a word into a sentiment label that reflects the net emotion that the word evokes.

We refer to the general setting of learning from both labels on examples and features as *dual supervision*. This setting arises more broadly in tasks where in addition to labeled documents, it is frequently possible to provide domain knowledge in the form of words, or phrases (Zaidan and Eisner, 2008) or even more sophisticated linguistic features, that associate strongly with a class. Recent work (Druck et al., 2008; Sindhwani and Melville, 2008) has demonstrated that the presence of word supervision can greatly reduce the number of labeled documents

Proceedings of the NAACL HLT Workshop on Active Learning for Natural Language Processing, pages 49–57,
Boulder, Colorado, June 2009.

required to build high quality text classifiers.

In general, these two sources of supervision are not mutually redundant, and have different annotation costs, human response quality, and degrees of utility towards learning a dual supervision model. This leads naturally to the problem of *active dual supervision*, or, how to optimally query a human oracle to simultaneously collect document *and* feature annotations, with the objective of building the highest quality model with the lowest cost. Much of the machine learning literature on active learning has focused on one-sided example-only annotation for classification problems. Less attention has been devoted to simultaneously acquiring alternative forms of supervisory domain knowledge, such as the kind routinely encountered in NLP. Our contribution may be viewed as a step in this direction.

2 Dual supervision

Most work in supervised learning has focused on learning from examples, each represented by a set of feature values and a class label. In dual supervision we consider an additional aspect, by way of labels of features, which convey prior knowledge on associations of features to particular classes. Since we deal only with text classification in this paper, all features represent term-frequencies of words, and as such we use *feature* and *word* interchangeably.

The active learning schemes we explore in this paper are broadly applicable to any learner that can support dual supervision, but here we focus on active learning for the Pooling Multinomials classifier (Melville et al., 2009) described below. In concurrent related work, we propose active dual supervision schemes for a class of graph-based and kernel-based dual supervision methods (Sindhwani et al., 2009).

2.1 Pooling Multinomials

The Pooling Multinomials classifier was introduced by Melville et al. (2009) as an approach to incorporate prior lexical knowledge into supervised learning for better sentiment detection. In the context of sentiment analysis, lexical knowledge is available in terms of the prior sentiment-polarity of words. From a dual supervision point of view, this knowledge can be seen as labeled features, since the lexicon effectively provides associations of a set of words with the positive or negative class.

Pooling Multinomials classifies unlabeled examples just as in multinomial Naïve Bayes classification (McCallum and Nigam, 1998), by predicting the class with the maximum likelihood, given by $argmax_{c_j} P(c_j) \prod_i P(w_i|c_j)$; where $P(c_j)$ is the prior probability of class c_j, and $P(w_i|c_j)$ is the probability of word w_i appearing in a document of class c_j. In the absence of background knowledge about the class distribution, we estimate the class priors $P(c_j)$ solely from the training data. However, unlike regular Naïve Bayes, the conditional probabilities $P(w_i|c_j)$ are computed using both the labeled examples and the labeled features.

Pooling distributions is a general approach for combining information from multiple sources or experts; where experts are typically represented in terms of probability distributions (Clemen and Winkler, 1999). Here, we only consider the special case of combining multinomial distributions from two sources – namely, the labeled examples and labeled features. The multinomial parameters of such models can be easily combined using the *linear opinion pool* (Clemen and Winkler, 1999), in which the aggregate probability is given by $P(w_i|c_j) = \alpha P_e(w_i|c_j) + (1 - \alpha) P_f(w_i|c_j)$; where $P_e(w_i|c_j)$ and $P_f(w_i|c_j)$ represent the probability assigned by using the example labels and feature labels respectively, and α is the weight for combining these distributions. The weight indicates a level of confidence in each source of information, and Melville et al. (2009) explore ways of automatically selecting this weight. However, in order to not confound our results with the choice of weight-selection mechanism, here we make the simplifying assumption that the two experts based on instance and feature labels are equally valuable, and as such set α to 0.5.

To learn a model from the labeled examples we compute conditionals $P_e(w_i|c_j)$ based on observed term frequencies, as in standard Naïve Bayes classification. In addition, for Pooling Multinomials we need to construct a multinomial model representing the labeled features in the background knowledge. For this, we assume that the feature-class associations provided by labeled features are implicitly arrived at by human experts by examining many positive and negative sentiment documents. So we

attempt to select the parameters $P_f(w_i|c_j)$ of the multinomial distributions that would generate such documents. The exact values of these conditionals are presented below. Their derivation is not directly pertinent to the subject of this paper, but can be found in (Melville et al., 2009).

Given:
$\mathcal{V}$ – the vocabulary, i.e., set of words in our domain
$\mathcal{P}$ – set of words labeled as positive
$\mathcal{N}$ – set of words labeled as negative
$\mathcal{U}$ – set of unknown words, i.e. $\mathcal{V} - (\mathcal{N} \cup \mathcal{P})$
m – size of vocabulary, i.e. $|\mathcal{V}|$
p – number of positive words, i.e. $|\mathcal{P}|$
n – number of negative words, i.e. $|\mathcal{N}|$

All words in the vocabulary can be divided into three categories – words with a positive label, negative label, and unknown label. We refer to the probability of any positive term appearing in a positive document simply as $P_f(w_+|+)$. Similarly, we refer to the probability of any negative term appearing in a negative document as $P_f(w_-|-)$; and the probability of an unknown word in a positive or negative context as $P_f(w_u|+)$ and $P_f(w_u|-)$ respectively. The generative model for labeled features can then be defined by:

$$P_f(w_+|+) = P_f(w_-|-) = \frac{1}{p+n}$$
$$P_f(w_+|-) = P_f(w_-|+) = \frac{1}{p+n} \times \frac{1}{r}$$
$$P_f(w_u|+) = \frac{n(1-1/r)}{(p+n)(m-p-n)}$$
$$P_f(w_u|-) = \frac{p(1-1/r)}{(p+n)(m-p-n)}$$

where, the *polarity level*, r, is a measure of how much more likely it is for a positive term to occur in a positive document compared to a negative term. The value of r is set to 100 in our experiments, as done in (Melville et al., 2009).

2.2 Learning from example vs. feature labels

Dual supervision makes it possible to learn from labeled examples and labeled features simultaneously; and, as in most supervised learning tasks, one would expect more labeled data of either form to lead to more accurate models. In this section we explore the influence of increased number of instance labels and feature labels independently, and also in tandem.

For these, and all subsequent experiments, we use 10-fold cross-validation on the publicly available data of movie reviews provided by Pang et al. (2002). This data consists of 1000 positive and 1000 negative reviews from the Internet Movie Database; where positive labels were assigned to reviews that had a rating above 3.5 stars and negative labels were assigned to ratings of 2 stars and below. We use a bag-of-words representation of reviews, where each review is represented by the term frequencies of the 5000 most frequent words across all reviews, excluding stop-words.

In order to study the effect of increasing number of labels we need to simulate a human oracle labeling data. In the case of examples this is straightforward, since all examples in the *Movies* dataset have labels. However, in the case of features, we do not have a gold-standard set of feature labels. So in order to simulate human responses to queries for feature labels, we construct a *feature oracle* in the following manner. The information gain of words with respect to the known true class labels in the dataset is computed using binary feature representations. Next, out of the 5000 total words, the top 1000 as ranked by information gain are assigned a label. This label is the class in which the word appears more frequently. The oracle returns a "dont know" response for the remaining words. Thus, this oracle simulates a human domain expert who is able to recognize and label the most relevant task-specific words, and also reject a word that falls below the relevance threshold. For instance, in sentiment classification, we would expect a "don't know" response for non-polar words.

We ran experiments beginning with a classifier provided with labels for 10 randomly selected instances and 10 randomly selected features. We then compare three schemes - Instances-then-features, Features-then-instances, and Passive Interleaving. As the name suggests, *Instances-then-features*, is provided labels for randomly selected instances until all instances have been labeled, and then switches to labeling features. Similarly, *Features-then-instances* acquires labels for randomly selected features first and then switches to getting instance labels. In *Passive Interleaving* we probabilistically switch be-

tween issuing queries for randomly chosen instance and feature labels. In particular, at each step we choose to query for an instance with probability 0.36, otherwise we query for a feature label. The instance-query rate of 0.36 is selected based on the ratio of available instances (1800) to available features (5000). The results of these learning curves are presented in Fig. 1. Note that the x-axis in the figure corresponds to the number of queries issued. As discussed earlier, in the case of features, the oracle may respond to a query with a class label or may issue a "don't know" response, indicating that no label is available. As such, the number of feature-queries on the x-axis does not correspond to the number of actual known feature labels. We would expect that on average 1 in 5 feature-label queries prompts a response from the feature oracle that results in a known feature label being provided.

At the end of the learning curves, each method has labels for all available instances and features; and as such, the last points of all three curves are identical. The results show that fixing the number of labeled features, and increasing the number of labeled instances steadily improves classification accuracy. This is what one would expect from traditional supervised learning curves. More interestingly, the results also indicate that we can fix the number of instances, and improve accuracy by labeling more features. Finally, results on Passive Interleaving show that, though both feature labels and example labels are beneficial by themselves, dual supervision which exploits the interaction of examples and features does in fact benefit from acquiring both types of labels concurrently.

For all results above, we are selecting instances and/or features to be labeled uniformly at random. Based on previous work in active learning one would expect that we can select instances to be labeled more efficiently, by having the learner decide which instances it is most likely to benefit from. The results in this section suggests that actively selecting features to be labeled may also be beneficial. Furthermore, the Passive Interleaving results suggest that an ideal active dual supervision scheme would actively select both instances and features for labeling. We begin by exploring active learning for feature labels in the next section, and then consider the simultaneous selection of instances and features in Sec. 4.

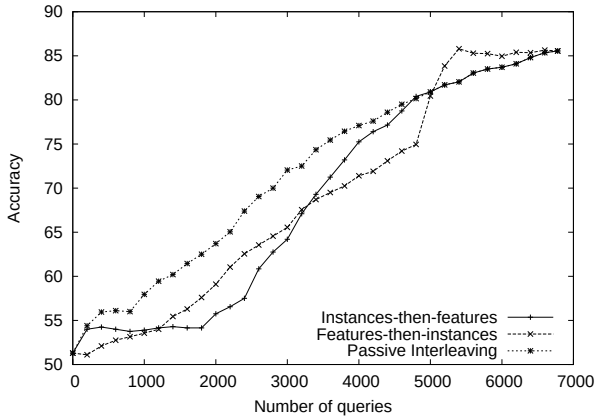


Figure 1: Comparing the effect of instance and feature label acquisition in dual supervision.

3 Acquiring feature labels

Traditional active learning has primarily focused on selecting unlabeled *instances* to be labeled. The dual-supervision setting now provides us with an additional dimension to active learning, where labels may also be acquired for features. In this section we look at the novel task of active learning applied only to feature-label acquisition. In Section 4 we study the more general task of active dual supervision, where both instance and feature labels may be acquired concurrently.

3.1 Feature uncertainty vs. certainty

A very common approach to active learning for instances is Uncertainty Sampling (Lewis and Catlett, 1994). In this approach we acquire labels for instances that the current model is most uncertain about. Uncertainty Sampling is founded on the heuristic that uncertain instances are close to the current classification boundary, and acquiring the correct labels for them are likely to help refine the location of this boundary. Despite its simplicity, Uncertainty Sampling is usually quite effective in practice; which raises the question of whether one can apply the same principle to feature-label acquisition. In this case, we want to select unlabeled features that the current model is most uncertain about.

Much like instance uncertainty, feature uncertainty can be measured in different ways, depending on the underlying method used for dual supervision. For instance, if the learner produces a lin-

ear classifier as in (Sindhwani and Melville, 2008), we could use the magnitude of the weights on the features as a measure of uncertainty – where lower weights indicate less certainty. Since Pooling Multinomials builds a multinomial Naïve Bayes model, we can directly use the model's conditional probabilities of each word (feature) given a class.

For ease of exposition we refer to the two classes in binary classification as *postive* (+) and *negative* (-), without loss of generality. Given the probabilities of word f belonging to the positive and negative class, $P(f|+)$ and $P(f|-)$, we can determine the uncertainty of a feature using the absolute value of the log-odds ratio, i.e.,

$$abs\left(\log\left(\frac{P(f|+)}{P(f|-)}\right)\right) \quad (1)$$

The smaller this value, the more uncertain the model is about the feature's class association. In every iteration of active learning we can select the features with the lowest certainty scores. We refer to this approach as *Feature Uncertainty*.

Though Uncertainty Sampling for features seems like an appealing notion, it may not lead to better models. If a classifier is uncertain about a feature, it may have insufficient information about this feature and may indeed benefit from learning its label. However, it is also quite likely that a feature has a low certainty score because it does not carry much discriminative information about the classes. In the context of sentiment detection, one would expect that neutral/non-polar words will appear to be uncertain words. For example, words such as "the" which are unlikely to help in discriminating between classes, are also likely to be considered the most uncertain. As we shortly report, on the movies dataset, Feature Uncertainty ends up wasting queries on such words ending up with performance inferior to random feature queries. What works significantly better is an alternative strategy which acquires labels for features in the descending order of the score in Eq 1. We refer to this approach as *Feature Certainty*.

3.2 Expected feature utility

The intuition underlying the feature certainty heuristic is that it serves to confirm or correct the orientation of model probabilities on different words during the active learning process. One can argue that feature certainty is also suboptimal in that queries may be wasted simply confirming confident predictions, which is of limited utility to the model. An alternative to using a certainty-based heuristic, is to directly estimate the expected value of acquiring each feature label. Such Expected Utility (Estimated Risk Minimization) approaches have been applied successfully to traditional active learning (Roy and McCallum, 2001), and to active feature-value acquisition (Melville et al., 2005). In this section we describe how this Expected Utility framework can be adapted for feature-label acquisition.

At every step of active learning for features, the next best feature to label is one that will result in the highest improvement in classifier performance. Since the true label of the unlabeled features are unknown prior to acquisition, it is necessary to estimate the potential impact of every feature query for all possible outcomes.[1] Hence, the decision-theoretic optimal policy is to ask for feature labels which, once incorporated into the data, will result in the highest increase in classification performance in *expectation*.

If f_j is the label of the j-th feature, and q_j is the query for this feature's label, then the Expected Utility of a feature query q_j can be computed as:

$$EU(q_j) = \sum_{k=1}^{K} P(f_j = c_k)\mathcal{U}(f_j = c_k) \quad (2)$$

Where $P(f_j = c_k)$ is the probability that f_j will be labeled with class c_k, and $\mathcal{U}(f_j = c_k)$ is the utility to the model of knowing that f_j has the label c_k. In practice, the true values of these two quantities are unknown, and the main challenge of any Expected Utility approach is to accurately estimate these quantities from the data currently available.

A direct way to estimate the utility of a feature label to classification, is to measure classification accuracy on the training set of a model built using this feature label. However, small changes in the model that may result from a acquiring a single additional feature label may not be reflected by a change in accuracy. As such, we use a more fine-grained measure of classifier performance, Log Gain, which is

[1] In the case of binary polarity classification, the possible outcomes are a *positive* or *negative* label for a queried feature.

computed as follows. For a model induced from a training set T, let $\hat{P}(c_k|x_i)$ be the probability estimated by the model that instance x_i belongs to class c_k; and $\mathbb{I}$ is an indicator function such that $\mathbb{I}(c_k, x_i) = 1$ if c_k is the correct class for x_i and $\mathbb{I}(c_k, x_i) = 0$, otherwise. Log Gain is then defined as:

$$LG(x_i) = -\sum_{k=1}^{K} \mathbb{I}(c_k)\hat{P}(c_k|x_i) \quad (3)$$

Then the utility of a classifier, $\mathcal{U}$, can be measured by summing the Log Gain for all instances in the training set T. A lower value of Log Gain indicates a better classifier performance. For a deeper discussion of this measure see (Saar-Tsechansky et al., 2008).

In Eq. 2, apart from the measure of utility, we also do not know the true probability distribution of labels for the feature under consideration. This too can be estimated from the training data, by seeing how frequently the word appears in documents of each class. In a multinomial Naïve Bayes model we already collect these statistics in order to determine the conditional probability of a class given a word, i.e. $P(f_j|c_k)$. We can use these probabilities to get an estimate of the feature label distribution, $\hat{P}(f_j = c_k) = \frac{P(f_j|c_k)}{\sum_{k=1}^{K} P(f_j|c_k)}$.

Given the estimated values of the feature-label distribution and the utility of a particular feature query outcome, we can now estimate the Expected Utility of each unknown feature, and select the features with the highest Expected Utility to modeling.

Though theoretically appealing, this approach is quite computationally intensive if applied to evaluate all unknown features. In the worst case it requires building and evaluating models for each possible outcome of each unlabeled feature query. If you have m features and K classes, this approach requires training $O(mK)$ classifiers. However, the complexity of the approach can be significantly alleviated by only applying Expected Utility evaluation to a sub-sample of all unlabeled features. Given the large number of features with no true class labels, selecting a sample of available features uniformly at random may be sub-optimal. Instead we select a sample of features based on Feature Certainty. In particular we select the top 100 unknown features that the current model is most certain about, and identify the features in this pool with the highest Expected Utility. We refer to this approach as *Expected Feature Utility*. We use Feature Certainty to sub-sample the available feature queries, since this approach is more likely to select features for which the label is known by the Oracle.

3.3 Active learning with feature labels

We ran experiments comparing the three different active learning approaches described above. In these, and all subsequent experiments, we begin with a model trained on 10 labeled features and 100 labeled instances, which were randomly selected. From our prior efforts of manually labeling such data, we find this to be a reasonable initial setting.

The experiments in this section focus only on the selection of *features* to be labeled. So, in each iteration of active learning we select the next 10 feature-label queries, based on Feature Uncertainty, Feature Certainty, or Expected Feature Utility. As a baseline, we also compare to the performance of a model that selects features uniformly at random. Our results are presented in Fig. 2.

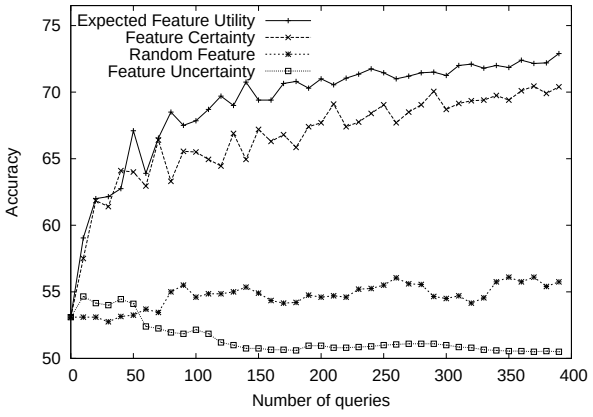


Figure 2: Comparing different active learning approaches for acquiring feature labels.

The results show that Feature Uncertainty, which is a direct analog of Uncertainty Sampling, actually performs worse than random sampling. Many uncertain features may actually not be very useful in discriminating between the classes, and selecting them can be systematically worse than selecting uniformly at random. However, the converse approach

of Feature Certainty does remarkably well. This may be because polarized words are better for learning, but it is also likely that querying for such words increases the likelihood of selecting one whose label is known to the oracle.

The results on Expected Feature Utility show that estimating the expected impact of potential labels for features does in fact perform much better than feature certainty. The results confirm that despite our crude estimations in Eq. 2, Expected Feature Utility is an effective approach to active learning of feature labels. Furthermore, we demonstrate that by applying the approach to only a small sub-sample of certain features, we are able to make this method computationally feasible to use in practice. Increasing the size of the sample of candidate feature queries is likely to improve performance, at the cost of increased time in selecting queries.

4 Active dual supervision

In the previous section we demonstrated that actively selecting informative features to be labeled is significantly better than random selection. In this section, we look at the complementary task of selecting instances to be labeled, and combined active learning for both forms of supervision.

Selecting unlabeled examples for learning has been a well-studied problem, and we use Uncertainty Sampling (Lewis and Catlett, 1994), which has been shown to be a computationally efficient and effective approach in the literature. In particular we select unlabeled examples to be labeled in order of decreasing uncertainty, where uncertainty is measured in terms of the margin, as done in (Melville and Mooney, 2004). The margin on an unlabeled example is defined as the absolute difference between the class probabilities predicted by the classifier for the given example, i.e., $|P(+|x) - P(-|x)|$. We refer to the selection of instances based on this uncertainty as Instance Uncertainty, in order to distinguish it from Feature Uncertainty.

We ran experiments as before, comparing selection of instances using Instance Uncertainty and selection of features using Expected Feature Utility. In addition, we also combine these to methods by interleaving feature and instance selection. In particular, we first order instances in decreasing order of uncertainty, and features in terms of decreasing Expected Feature Utility. We then probabilistically select instances or features from the top of these lists, where, as before, the probability of selecting an instance is 0.36. Recall that this probability corresponds to the ratio of available instances (1800) and features (5000). We refer to this approach as Active Interleaving, in contrast to Passive Interleaving, which we also present as a baseline. Recall that Passive Interleaving corresponds to probabilistically interleaving queries for randomly chosen, not actively chosen, examples and features. Our results are presented in Fig. 3.

We observe that, Instance Uncertainty performs better than Passive Interleaving, which in turn is better than random selection of only instances or features – as seen in Fig. 1. However, effectively selecting features labels, via Expected Feature Utility, does even better than actively selecting only instances. Finally, selecting instance and features simultaneously via Active Interleaving performs better than active learning of features or instances separately. Active Interleaving is indeed very effective, reaching an accuracy of 77% with only 500 queries, while Passive Interleaving requires more than 4000 queries to reach the same performance – as evidenced by Fig. 1

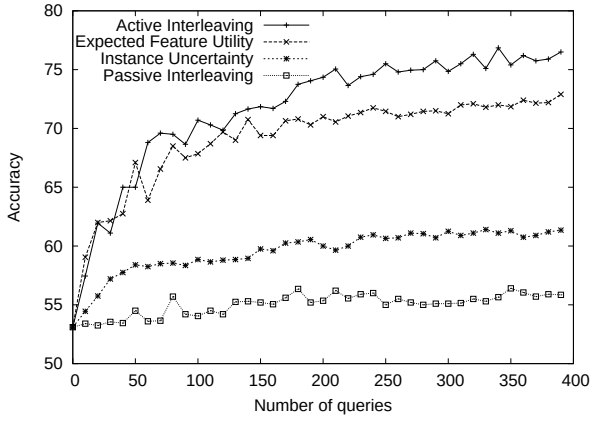


Figure 3: Comparing Active Interleaving to alternative label acquisition strategies.

5 Related work

Active learning in the context of dual supervision models is a new area of research with very little prior

work, to the best of our knowledge. Most prior work has focused on pooled-based active learning, where examples from an unlabeled pool are selected for labeling (Cohn et al., 1994; Tong and Koller, 2000). In contrast, active feature-value acquisition (Melville et al., 2005) and *budgeted learning* (Lizotte et al., 2003) focus on estimating the value of acquiring missing features, but do not deal with the task of learning from feature *labels*. In contrast, Raghavan and Allan (2007) and Raghavan et al. (2006) study the problem of *tandem learning* where they combine uncertainty sampling for instances along with co-occurence based interactive feature selection. Godbole et al. (2004) propose notions of feature uncertainty and incorporate the acquired feature labels, into learning by creating one-term mini-documents.

Learning from labeled examples and features via dual supervision, is itself a new area of research. Sindhwani et al. (2008) use a kernel-based framework to build dual supervision into co-clustering models. Sindhwani and Melville (2008) apply similar ideas for graph-based sentiment analysis. There have also been previous attempts at using only feature supervision, mostly along with unlabeled documents. Much of this work (Schapire et al., 2002; Wu and Srihari, 2004; Liu et al., 2004; Dayanik et al., 2006) has focused on using labeled features to generate *pseudo-labeled examples* that are then used with well-known models. In contrast, Druck et al. (2008) constrain the outputs of a multinomial logistic regression model to match certain reference distributions associated with labeled features.

6 Perspectives and future work

Though Active Interleaving is a very effective approach to active dual supervision, there is still a lot of room for improvement. Firstly, Active Interleaving relies on Uncertainty Sampling for the selection of instances. Though Uncertainty Sampling has the advantage of being fast and effective, there exist approaches that lead to better models with fewer examples – usually at the cost of computation time. One such method, estimating error reduction (Roy and McCallum, 2001), is a direct analog of Expected Feature Utility applied to instance selection. One would expect that an improvement in instance selection, should directly improve any method that combines instance and feature label selection. Secondly, Active Interleaving uses the simple approach of probabilistically choosing to select an instance or feature for each subsequent query. However, a more intelligent active scheme should be able to assess if an instance or feature would be more beneficial at each step. Furthermore, we do not currently consider the cost of acquiring labels. Presumably labeling a feature versus labeling an instance could incur very different costs – which could be monetary costs or time taken for each annotation. Fortunately, the Expected Utility method is very flexible, and allows us to address all these issues within a single framework. We can specifically estimate the expected utility of different forms of annotation, per unit cost. For instance, Provost et al. (2007) use such an approach to estimate the utility of acquiring class labels and feature values (not labels) per unit cost, within one unified framework. A similar method can be applied for a holistic approach to active dual supervision, where the Expected Utility of an instance or feature label query q, can be computed as $EU(q) = \sum_{k=1}^{K} P(q = c_k) \frac{\mathcal{U}(q=c_k)}{\omega_q}$; where ω_q is cost of the query q, and utility $\mathcal{U}$ can be computed as in Eq. 3. By evaluating instances and features on the same scale, and by measuring utility per unit cost of acquisition, such a framework should enable us to handle the trade-off between the costs and benefits of the different types of acquisitions. The primary challenge in the success of this approach is to *accurately* and *efficiently* estimate the different quantities in the equation above, using only the training data currently available. These are directions for future exploration.

7 Conclusions

This paper is a preliminary foray into active dual supervision. We have demonstrated that not only is combining example and feature labels beneficial for modeling, but that actively selecting the most informative examples and features for labeling can significantly reduce the burden of annotating such data. In future work, we would like to explore more effective solutions to the problem, and also to corroborate our results on a larger number of datasets and under different experimental settings.

References

R. T. Clemen and R. L. Winkler. 1999. Combining probability distributions from experts in risk analysis. *Risk Analysis*, 19:187–203.

D. Cohn, L. Atlas, and R. Ladner. 1994. Improving generalization with active learning. *Machine Learning*, 15(2):201–221.

Aynur Dayanik, David D. Lewis, David Madigan, Vladimir Menkov, and Alexander Genkin. 2006. Constructing informative prior distributions from domain knowledge in text classification. In *SIGIR*.

G. Druck, G. Mann, and A. McCallum. 2008. Learning from labeled features using generalized expectation criteria. In *SIGIR*.

S. Godbole, A. Harpale, S. Sarawagi, and S. Chakrabarti. 2004. Document classification through interactive supervision of document and term labels. In *PKDD*.

David D. Lewis and Jason Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Proc. of 11th Intl. Conf. on Machine Learning (ICML-94)*, pages 148–156, San Francisco, CA, July. Morgan Kaufmann.

Bing Liu, Xiaoli Li, Wee Sun Lee, and Philip Yu. 2004. Text classification by labeling words. In *AAAI*.

Dan Lizotte, Omid Madani, and Russell Greiner. 2003. Budgeted learning of naive-Bayes classifiers. In *UAI*.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive Bayes text classification. In *AAAI Workshop on Text Categorization*.

Prem Melville and Raymond J. Mooney. 2004. Diverse ensembles for active learning. In *Proc. of 21st Intl. Conf. on Machine Learning (ICML-2004)*, pages 584–591, Banff, Canada, July.

Prem Melville, Maytal Saar-Tsechansky, Foster Provost, and Raymond Mooney. 2005. An expected utility approach to active feature-value acquisition. In *ICDM*.

Prem Melville, Wojciech Gryc, and Richard Lawrence. 2009. Sentiment analysis of blogs by combining lexical knowledge with text classification. In *KDD*.

Bo Pang and Lilian Lee. 2008. *Opinion mining and sentiment analysis*. Foundations and Trends in Information Retrieval: Vol. 2: No 1, pp 1-135.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *EMNLP*.

Foster Provost, Prem Melville, and Maytal Saar-Tsechansky. 2007. Data acquisition and cost-effective predictive modeling: Targeting offers for electronic commerce. In *ICEC '07: Proceedings of the ninth international conference on Electronic commerce*.

Hema Raghavan, Omid Madani, and Rosie Jones. 2006. Active learning with feedback on features and instances. *J. Mach. Learn. Res.*, 7:1655–1686.

H. Raghavan, O. Madani, and R. Jones. 2007. An interactive algorithm for asking and incorporating feature feedback into support vector machines. In *SIGIR*.

Nicholas Roy and Andrew McCallum. 2001. Toward optimal active learning through sampling estimation of error reduction. In *ICML*.

Maytal Saar-Tsechansky, Prem Melville, and Foster Provost. 2008. Active feature-value acquisition. In *Management Science*.

Robert E. Schapire, Marie Rochery, Mazin G. Rahim, and Narendra Gupta. 2002. Incorporating prior knowledge into boosting. In *ICML*.

Vikas Sindhwani and Prem Melville. 2008. Document-word co-regularization for semi-supervised sentiment analysis. In *ICDM*.

Vikas Sindhwani, Jianying Hu, and Alexandra Mojsilovic. 2008. Regularized co-clustering with dual supervision. In *NIPS*.

Vikas Sindhwani, Prem Melville, and Richard Lawrence. 2009. Uncertainty sampling and transductive experimental design for active dual supervision. In *ICML*.

Simon Tong and Daphne Koller. 2000. Support vector machine active learning with applications to text classification. In *Proc. of 17th Intl. Conf. on Machine Learning (ICML-2000)*.

Xiaoyun Wu and Rohini Srihari. 2004. Incorporating prior knowledge with weighted margin support vector machines. In *KDD*.

O. F. Zaidan and J. Eisner. 2008. Modeling annotators: A generative approach to learning from annotator rationales. In *EMNLP*.

Proactive Learning for Building Machine Translation Systems for Minority Languages

Vamshi Ambati
vamshi@cs.cmu.edu
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Jaime Carbonell
jgc@cs.cmu.edu
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA

Abstract

Building machine translation (MT) for many minority languages in the world is a serious challenge. For many minor languages there is little machine readable text, few knowledgeable linguists, and little money available for MT development. For these reasons, it becomes very important for an MT system to make best use of its resources, both labeled and unlabeled, in building a quality system. In this paper we argue that traditional active learning setup may not be the right fit for seeking annotations required for building a Syntax Based MT system for minority languages. We posit that a relatively new variant of active learning, Proactive Learning, is more suitable for this task.

1 Introduction

Speakers of minority languages could benefit from fluent machine translation (MT) between their native tongue and the dominant language of their region. But scarcity in capital and know-how has largely restricted machine translation to the dominant languages of first world nations. To lower the barriers surrounding MT system creation, we must reduce the time and resources needed to develop MT for new language pairs. Syntax based MT has proven to be a good choice for minority language scenario (Lavie et al., 2003). While the amount of parallel data required to build such systems is orders of magnitude smaller than corresponding phrase based statistical systems (Koehn et al., 2003), the variety of linguistic annotation required is greater. Syntax based MT systems require lexicons that provide coverage for the target translations, synchronous grammar rules that define the divergences in word-order across the language-pair. In case of minority languages one can only expect to find meagre amount of such data, if any. Building such resources effectively, within a constrained budget, and deploying an MT system is the need of the day.

We first consider 'Active Learning' (AL) as a framework for building annotated data for the task of MT. However, AL relies on unrealistic assumptions related to the annotation tasks. For instance, AL assumes there is a unique omniscient oracle. In MT, it is possible and more general to have multiple sources of information with differing reliabilities or areas of expertise. A literate bilingual speaker with no extra training can produce translations for word, phrase or sentences and even align them. But it requires a trained linguist to produce syntactic parse trees. AL also assumes that the single oracle is perfect, always providing a correct answer when requested. In reality, an oracle (human or machine) may be incorrect (fallible) with a probability that should be a function of the difficulty of the question. There is also no notion of cost associated with the annotation task, that varies across the input space. But in MT, it is easy to see that length of a sentence and cost of translation are superlinear. Also not all annotation tasks for MT have the same level of difficulty or cost. For example, it is relatively cheap to ask a bilingual speaker whether a word, phrase or sentence was correctly translated by the system, but a bit more expensive to ask for a correction. Assumptions like these render active learning unsuit-

Proceedings of the NAACL HLT Workshop on Active Learning for Natural Language Processing, pages 58–61,
Boulder, Colorado, June 2009.

able for our task at hand which is building an MT system for languages with limited resources. We make the case for "Proactive Learning" (Donmez and Carbonell, 2008) as a solution for this scenario.

In the rest of the paper, we discuss syntax based MT approach in Section 2. In Section 3 we first discuss active learning approaches for MT and detail the characteristics of MT for minority languages problem that render traditional active learning unsuitable for practical purposes. In Section 4 we discuss proactive learning as a potential solution for the current problem. We conclude with some challenges that still remain in applying proactive learning for MT.

2 Syntax Based Machine Translation

In recent years, corpus based approaches to machine translation have become predominant, with Phrase Based Statistical Machine Translation (PB-SMT) (Koehn et al., 2003) being the most actively progressing area. Recent research in syntax based machine translation (Yamada and Knight, 2001; Chiang, 2005) incorporates syntactic information to ameliorate the reordering problem faced by PB-SMT approaches. While traditional approaches to syntax based MT were dependent on availability of manual grammar, more recent approaches operate within the resources of PB-SMT and induce hierarchical or linguistic grammars from existing phrasal units, to provide better generality and structure for reordering (Yamada and Knight, 2001; Chiang, 2005; Wu, 1997).

2.1 Resources for Syntax MT

Syntax based approaches to MT seek to leverage the structure of natural language to automatically induce MT systems. Depending upon the MT system and the paradigm, the resource requirements may vary and could also include modules such as morphological analyzers, sense disambiguation modules, generators etc. A detailed discussion of the comprehensive pipeline, may be out of the scope of this paper, more so because such resources can not be expected in a low-resource language scenario. We only focus on the quintessential set of modules for MT pipeline - data acquisition, word-alignment, syntactic analysis etc. The resources can broadly be categorized as 'monolingual' vs 'bilingual' depending upon whether it requires knowledge in one language or both languages for annotation. A sample of the different kinds of data and annotation that is expected by an MT system is shown below. Each of the additional information can be seen as extra annotations for the 'Source' sentence. The language of target in the example is 'Hindi'.

- **Source:** John ate an apple
- **Target:** John ne ek seb khaya
- **Alignment:** (1,1),(2,5),(3,3),(4,4)
- **SourceParse:** (S (NP (NNP John)) (VP (VBD ate) (NP (DT an) (NN apple))))
- **Lexicon:** (seb → apple),(ate → khaya)
- **Grammar:** VP: V NP → NP V

3 Active Learning for MT

Modern syntax based MT rides on the success of both Statistical Machine Translation and Statistical Parsing. Active learning has been applied to Statistical Parsing (Hwa, 2004; Baldridge and Osborne, 2003) to improve sample selection for manual annotation. In case of MT, active learning has remained largely unexplored. Some attempts include training multiple statistical MT systems on varying amounts of data, and exploring a committee based selection for re-ranking the data to be translated and included for re-training. But this does not apply to training in a low-resource scenario where data is scarce.

In the rest of the section we discuss the different scenarios that arise in gathering of annotation for MT under a traditional 'active learning' setup and discuss the characteristics of the task that render it difficult.

3.1 Multiple Oracles

For each of the sub-tasks of annotation, in reality we have multiple sources of information or multiple oracles. We can elicit translations for building a parallel corpus from bilingual speakers who speak both the languages with certain accuracy or from a linguist who is well educated in the formal sense of the languages. With the success of collaborative sites like Amazon's 'Mechanical Turk' [1], one

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

can provide the task of annotation to multiple oracles on the internet (Snow et al., 2008). The task of word alignment can be posed in a similar fashion too. More interestingly, there are statistical tools like GIZA [2] that take as input un-annotated parallel data and propose automatic correspondences between words in the language-pair, giving scope to 'machine oracles'.

3.2 Varying Quality and Reliability

Oracles also vary on the correctness of the answers they provide (quality) as well as their availability (robustness) to answer. One typical distinction is 'human oracles' vs 'machine oracles'. Human oracle produce higher quality annotations when compared to a machine oracle. We would prefer a tree bank of parse trees that were manually created over automatically generated tree banks. Similar is the case with word-alignment and other tasks of translation. Some oracles are 'reluctant' to produce an output, for example parsers tend to break on really long sentences, but when they produce an output we can associate some confidence with it about the quality. One can expect a human oracle to produce parse trees for long sentences, but the quality could be questionable.

3.3 Non-uniform costs

Each of the annotation tasks has a non-uniform cost associated with it, the distribution of which is dependent upon the difficulty over the input space. Clearly, length of the sentence is a good indicator of the cost. It takes much longer to translate a sentence of 100 words than to translate one with 10 words. It takes at least twice as long to create word-alignment correspondences for a sentence-pair with 40 tokens than a pair with 20 tokens. Similarly, a human takes much longer to manually create parse tree for a long sentence than a short sentence.

It is also the case that not all oracles have the same non-uniform cost distribution over the input space. Some oracles are more expensive than the others. For example a practicing linguist's time is perhaps costlier than that of an undergraduate who is a bilingual speaker. As noticed above, this may reflect upon the quality of annotation for the task, but sometimes a tradeoff to make is cost vs quality. We can not afford to introduce a grammar rule of low-quality into the system, but can possibly do away with an incorrect word-correspondence link.

[2] http://www.fjoch.com/GIZA++.html

4 Proactive Learning

Proactive learning (Donmez and Carbonell, 2008) is a generalization of active learning designed to relax unrealistic assumptions and thereby reach practical applications. Active learning seeks to select the most informative unlabeled instances and ask an omniscient oracle for their labels, so as to retrain the learning algorithm maximizing accuracy. However, the oracle is assumed to be infallible (never wrong), indefatigable (always answers), individual (only one oracle), and insensitive to costs (always free or always charges the same). Proactive learning relaxes all these four assumptions, relying on a decision-theoretic approach to jointly select the optimal oracle and instance, by casting the problem as a utility optimization problem subject to a budget constraint.

$$maximize\ E[V(S)]\ subject\ to\ B$$
$$max_{S \in UL} E[V(S)] - \lambda(\sum_k t_k * C_k) s.t$$
$$\sum_k t_k * C_k = B$$

The above equation can be interpreted as maximizing the expected value of labeling the input set S under the budget constraint B. The subscript k denotes the oracle from which the answer was elicited under a cost function C. A greedy approximation of the above results in the equation 1, where $E_k[V(x)]$ is the expected value of information of the example x corresponding to oracle k. One can design interesting functions that calculate $V(x)$ in case of MT. For example, selecting short sentences with an unresolved linguistic issue could maximize the utility of the data at a low cost.

$$(x*, k*) = argmax_{x \in U} E_k[V(x)]\ subject\ to\ B \quad (1)$$

We now turn to how proactive learning framework helps solve the issues raised for active learning in MT in section 3. We can address the issue of multiple oracles where one oracle is fallible or reluctant to answer, by factoring into Equation 2 its probability

function for returning an answer. The score returned by such a factoring can be called the utility associated with that input for a particular oracle. We call this $U(x, k)$. A similar factorization can be done in order to address the issue of oracles that are fallible.

$$\begin{aligned} U(x,k) &= P(ans|x,k) * V(x) - C_k \\ (x*,k*) &= argmax_{x \in U}\ U(x,k) \end{aligned}$$

Since we do not have the $P(ans/x, k)$ distribution information for each oracle, proactive learning proposes to discover this in a discovery phase under some allocated budget B_d. Once we have an estimate from the discovery phase, the rest of the labeling proceeds according to the optimization function. For more details of the algorithms refer (Donmez and Carbonell, 2008). Finally, we can also relax the assumption of uniform cost per annotation, but replacing the C_k term in the above equations with a $C_{non-unif_k}$ function denoting the non-uniform cost function associated with the oracle.

5 Future Challenges

While proactive learning is a good framework for building MT systems for minority languages, there are however a few issues that still remain that need careful attention.

Joint Utility: In a complex system like MT where different models combine forces to produce the translation we have a situation where we need to optimize not only for an input and the oracle, but also the kind of annotation we would like to elicit. For example given a particular translation model, we do not know if the most optimal thing at a given point is to seek more word-alignment annotation from a particular 'alignment oracle' or seek parse annotation from a 'parsing oracle'.

Machine oracles vs Human oracles: The assumption with an oracle is that the knowledge and expertise of the oracle does not change over the course of annotation. We do not assume that the oracle learns over time and hence the speed of annotation or perhaps the accuracy of annotation increases. This is however very common with 'machine oracles'. For example, an oracle that suggests automatic alignment of data using statistical concordances may initially be unreliable due to the less amount of data it is trained on, but as it receives more data, the estimates get better and so the system gets more reliable.

Evaluation: Performance of underlying system is typically done by well understood metrics like precision/recall. However, evaluation of MT output is quite subjective and automatic evaluation metrics may be too coarse to distinguish the nuances of translation. This becomes quite important in an online active learning setup, where we add annotated data incrementally, and the immediately trained translation models are not sufficient to make a difference in the scores of the evaluation metric.

References

Jason Baldridge and Miles Osborne. 2003. Active learning for hpsg parse selection. In *Proc. of the HLT-NAACL 2003*, pages 17–24, Morristown, NJ, USA. Association for Computational Linguistics.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. 43rd ACL*, pages 263–270, Morristown, NJ, USA. Association for Computational Linguistics.

Pinar Donmez and Jaime G. Carbonell. 2008. Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In *CIKM '08*, pages 619–628, New York, NY, USA. ACM.

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

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proc. of the HLT/NAACL*, Edomonton, Canada.

Alon Lavie, Stephan Vogel, Lori Levin, Erik Peterson, Katharina Probst, Ariadna Font Llitjós, Rachel Reynolds, Jaime Carbonell, and Richard Cohen. 2003. Experiments with a hindi-to-english transfer-based mt system under a miserly data scenario. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2(2):143–163.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Ng. 2008. Cheap and fast – but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the EMNLP 2008*, pages 254–263, Honolulu, Hawaii, October.

Dekai Wu. 1997. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403.

Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proc. of ACL '01*, pages 523–530, Morristown, NJ, USA. Association for Computational Linguistics.

Author Index