Efficient Annotation with the Jena ANnotation Environment (JANE)

Katrin Tomanek **Joachim Wermter** **Udo Hahn**
Jena University Language & Information Engineering (JULIE) Lab
Fürstengraben 30
D-07743 Jena, Germany
{tomanek|wermter|hahn}@coling-uni-jena.de

Abstract

With ever-increasing demands on the diversity of annotations of language data, the need arises to reduce the amount of efforts involved in generating such value-added language resources. We introduce here the Jena ANnotation Environment (JANE), a platform that supports the complete annotation life-cycle and allows for 'focused' annotation based on active learning. The focus we provide yields significant savings in annotation efforts by presenting only informative items to the annotator. We report on our experience with this approach through simulated and real-world annotations in the domain of immunogenetics for NE annotations.

1 Introduction

The remarkable success of machine-learning methods for NLP has created, for supervised approaches at least, a profound need for annotated language corpora. Annotation of language resources, however, has become a bottleneck since it is performed, with some automatic support (pre-annotation) though, by humans. Hence, annotation is a time-costly and error-prone process.

The demands for annotated language data is increasing at different levels. After the success in syntactic (Penn TreeBank (Marcus et al., 1993)) and propositional encodings (Penn PropBank (Palmer et al., 2005)), more sophisticated semantic data (such as temporal (Pustejovsky et al., 2003) or opinion annotations (Wiebe et al., 2005)) and discourse data (e.g., for anaphora resolution (van Deemter and Kibble, 2000) and rhetorical parsing (Carlson et al., 2003)) are being generated. Once the ubiquitous area of newswire articles is left behind, different domains (e.g., the life sciences (Ohta et al., 2002)) are yet another major concern. Furthermore, any new HLT application (e.g., information extraction, document summarization) makes it necessary to provide appropriate human annotation products. Besides these considerations, the whole field of non-English languages is desperately seeking to enter into enormous annotation efforts, at virtually all encoding levels, to keep track of methodological requirements imposed by such resource-intensive research activities.

Given this enormous need for high-quality annotations at virtually all levels the question turns up how to minimize efforts within an acceptable quality window. Currently, for most tasks several hundreds of thousands of text tokens (ranging between 200,000 to 500,000 text tokens) have to be scrutinized unless valid tagging judgments can be learned. While significant time savings have already been reported on the basis of automatic pre-tagging (e.g., for POS and parse tree taggings in the Penn TreeBank (Marcus et al., 1993), or named entity taggings for the Genia corpus (Ohta et al., 2002)), this kind of pre-processing does not reduce the number of text tokens actually to be considered.

We have developed the Jena ANnotation Environment (JANE) that allows to reduce annotation efforts by means of the *active learning* (AL) approach. Unlike random or sequential sampling of linguistic items to be annotated, AL is an intelligent selective

Proceedings of the Linguistic Annotation Workshop, pages 9–16,
Prague, June 2007.

sampling strategy that helps reduce the amount of data to be annotated substantially at almost no loss in annotation effectiveness. This is achieved by focusing on those items particularly relevant for the learning process.

In Section 2, we review approaches to annotation cost reduction. We turn in Section 3 to the description of JANE, our AL-based annotation system, while in Section 4 we report on the experience we made using the AL component in NE annotations.

2 Related Work

Reduction of efforts for training (semi-) supervised learners on annotated language data has always been an issue of concern. Semi-supervised learning provides methods to bootstrap annotated corpora from a small number of manually labeled examples. However, it has been shown (Pierce and Cardie, 2001) that semi-supervised learning is brittle for NLP tasks where typically large amounts of high quality annotations are needed to train appropriate classifiers.

Another approach to reducing the human labeling effort is *active learning* (AL) where the learner has direct influence on the examples to be manually labeled. In such a setting, those examples are taken for annotation which are assumed to be maximally useful for (classifier) training. AL approaches have already been tried for different NLP tasks (Engelson and Dagan, 1996; Hwa, 2000; Ngai and Yarowsky, 2000), though such studies usually report on simulations rather than on concrete experience with AL for real annotation efforts. In their study on AL for base noun phrase chunking, Ngai and Yarowsky (2000) compare the costs of rule-writing with (AL-driven) annotation to compile a base noun phrase chunker. They conclude that one should rather invest human labor in annotation than in rule writing.

Closer to our concerns is the study by Hachey et al. (2005) who apply AL to named entity (NE) annotation. There are some differences in the actual AL approach they chose, while their main idea, *viz.* to apply committee-based AL to speed up real annotations, is comparable to our work. They report on negative side effects of AL on the annotations and state that AL annotations are cognitively more difficult for the annotators to deal with (because the sentences selected for annotation are more complex). As a consequence, diminished annotation quality and higher per-sentence annotation times arise in their experiments. By and large, however, they conclude that AL selection should still be favored over random selection because the negative implications of AL are easily over-compensated by the significant reduction of sentences to be annotated to yield comparable classifier performance as under random sampling conditions.

Whereas Hatchey *et al.* focus only on one group of entity mentions (*viz.* four entity subclasses of the astrophysics domain), we report on broader experience when applying AL to annotate several groups of entity mentions in biomedical subdomains. We also address practical aspects as to how create the seed set for the first AL round and how one might estimate the efficiency of AL. The immense savings in annotation effort we achieve here (up to 75%) may mainly depend on the sparseness of many entity types in biomedical corpora. Furthermore, we here present a *general* annotation environment which supports AL-driven annotations for most segmentation problems, not just for NE recognition.

In contrast, annotation editors, such as e.g. WordFreak[1], typically offer facilities for supervised correction of automatically annotated text. This, however, is very different from the AL approach.

3 JANE – Jena ANnotation Environment

JANE, the Jena ANnotation Environment, supports the whole annotation life-cycle including the compilation of annotation projects, annotation itself (via an external editor), monitoring, and the deployment of annotated material. In JANE, an *annotation project* consists of a *collection of documents* to be annotated, an associated *annotation schema* – a specification of what has to be annotated in which way, according to the accompanying annotation guidelines – a set of configuration parameters, and an *annotator* assigned to it.

We distinguish two kinds of annotation projects: A *default project*, on the one hand, contains a predefined and fixed collection of naturally occurring documents which the annotator handles independently of each other. In an *active learning project*, on the other hand, the annotator has access to exactly one

[1] `http://wordfreak.sourceforge.net`

(AL-computed pseudo) document at a time. After such a document has completely been annotated, a new one is dynamically constructed which contains those sentences for annotation which are the most informative ones for training a classifier. Besides annotators who actually do the annotation, there are *administrators* who are in charge of (annotation) project management, monitoring the annotation progress, and deployment, i.e., exporting the data to other formats.

JANE consists of one central component, the *annotation repository*, where all annotation and project data is stored centrally, two *user interfaces*, namely one for the annotators and one for the administrator, and the *active learning* component which interactively generates documents to speed up the annotation process. All components communicate with the annotation repository through a network socket – allowing JANE to be run in a distributed environment. JANE is largely platform-independent because all components are implemented in Java. A test version of JANE may be obtained from `http://www.julielab.de`.

3.1 Active Learning Component

One of the most established approaches to active learning is based on the idea to build an ensemble of classifiers from the already annotated examples. Each classifier then makes its prediction on all unlabeled exampels. Examples on which the classifiers in the ensemble disagree most in their predictions are considered informative and are thus requested for labeling. Obviously, we can expect that adding these examples to the training corpus will increase the accuracy of a classifier trained on this data (Seung et al., 1992). A common metric to estimate the disagreement within an ensemble is the so-called *vote entropy*, the entropy of the distribution of labels l_i assigned to an example e by the ensemble of k classifiers (Engelson and Dagan, 1996):

$$D(e) = -\frac{1}{\log k} \sum_{l_i} \frac{V(l_i, e)}{k} \log \frac{V(l_i, e)}{k}$$

Our AL component employs such an ensemble-based approach (Tomanek et al., 2007). The ensemble consists of $k = 3$ classifiers[2]. AL is run on the sentence level because this is a natural unit for many segmentation tasks. In each round, b sentences with the highest disagreement are selected.[3] The pool of (available) unlabeled examples can be very large for many NLP tasks; for NE annotations in the biomedical domain we typically download several hundreds of thousands of abstracts from PUBMED.[4] In order to avoid high selection times, we consider only a (random) subsample of the pool of unlabeled examples in each AL round. Both the selection size b (which we normally set to $b = 30$), the composition of the ensemble, and the subsampling ratio can be configured with the administration component.

AL selects single, non-contiguous sentences from *different* documents. Since the context of these sentences is still crucial for many (semantic) annotation decisions, for each selected sentence its original context is added (but blocked from annotation). When AL selection is finished, a new document is compiled from these sentences (including their contexts) and uploaded to the annotation repository. The annotator can then proceed with annotation.

Although optimized for NE annotations, the AL component may – after minor modifications of the feature sets being used by the classifiers – also be applied to other segmentation problems, such as POS or chunk annotations.

3.2 Administration Component

Administering large-scale annotation projects is a challenging management task for which we supply a GUI (Figure 1) to support the following tasks:

User Management Create accounts for administrators and annotators.

Creation of Projects The creation of an annotation project requires a considerable number of documents and other files (such as annotation schema definitions) to be uploaded to the annotation repository. Furthermore, several parameters, especially for AL projects have to be set appropriately.

Editing a Project The administrator can reset a project (especially when guidelines change, one

[2] Currently, we incorporate as classifiers Naive Bayes, Maximum Entropy, and Conditional Random Fields.

[3] Here, the vote entropy is calculated separately for each token. The sentence-level vote entropy is then the average over the respective token sequence.

[4] `http://www.ncbi.nlm.nih.gov/`

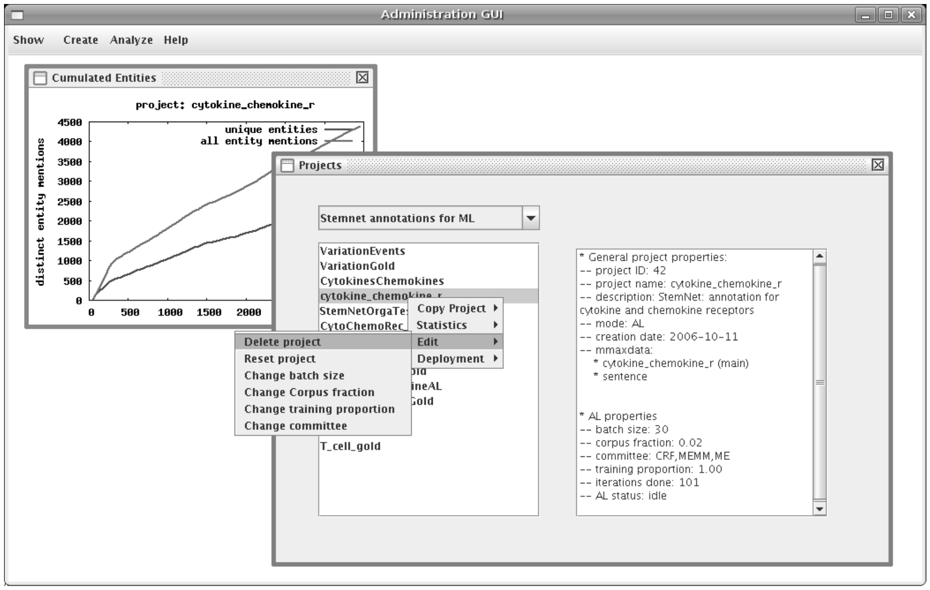

Figure 1: Administration GUI: frame in foreground shows actions that can be performed on an AL project.

might want to start the annotation process anew, i.e., delete all previous annotations but keep the rest of the project unchanged), delete a project, copy a project (which is helpful when several annotators label the same documents to check the applicability of the guidelines by inter-annotator agreement calculation), and change several AL-specific settings.

Monitoring the Annotation Process The administrator can check which documents of an annotation project have already been annotated, how long annotation took on the average, when an annotator logged in last time, etc. Furthermore, the progress of AL projects can be visualized by learning and disagreement curves and an enumeration of the number of (unique) entities found so far.

Inter-Annotator Agreement For related projects (projects sharing the same annotation schema and documents to be annotated) the degree to which several annotators mutually agree in their annotations can be calculated. Such an inter-annotator agreement (IAA) is common to estimate the quality and applicability of particular annotation guidelines (Kim and Tsujii, 2006). Currently, several IAA metrics of different strictness for NE annotations (and other segmentation tasks) are incorporated.

Deployment The annotation repository stores the annotations in a specific XML format (see Section 3.3). For deployment, the annotations may be needed in a different format. Currently, the administration GUI basically supports export into the IOB format. Only documents marked by the annotators as *'completely annotated'* are considered.

3.3 Annotation Component

As the annotators are rather domain experts (in our case graduate students of biology or related life sciences) than computer specialists, we wanted to make life for them as easy as possible. Hence, we provide a separate GUI for the annotators. After log-in the annotator is given an overview of his/her annotation projects along with a short description. Double clicking on a project, the annotators get a list with all documents in this project. Documents have different flags (*raw*, *in progress*, *done*) to indicate the current annotation state as set by each annotator.

Annotation itself is done with MMAX, an external annotation editor (Müller and Strube, 2003), which can be customized with respect to the particular annotation schema. The document to be annotated, the annotations, and the configuration parameters are stored in separate XML files. Our annotation repository reflects this MMAX-specific data structure.

Double clicking on a specific document directly opens MMAX for annotation. During annotation, the annotation GUI is locked to ensure data integrity. When working on an AL project, the annotator can start the AL selection process (which then runs on a separate high-performance machine) after having finished the annotation of the current document. During the AL selection process (it usually takes up to several minutes) the current project is blocked. However, meanwhile the annotator can go on annotating other projects.

3.4 Annotation Repository

The annotation repository is the heart of our annotation environment. All project, user, and annotation relevant data is stored here centrally. This is a crucial design criterion because it lets the administrator access (e.g., for backup or deployment) *all* annotations from one central site. Furthermore, the annotators do not have to care about how to shift the annotated documents to the managerial staff. All state information related to the entire annotation cycle is recorded and kept centrally in this repository.

The repository is realized as a relational database[5] reflecting largely the data structure of MMAX. Both, the GUIs and the AL component, communicate with the repository via the JDBC network driver. Thus, each component can be run on a different machine as long as it has a network connection to the annotation repository. This has two main advantages: First, annotators can work remotely (e.g., from home or from a physically dislocated lab). Second, resource-intensive tasks, e.g., AL selection, can be run on separate machines to which the annotators normally do not have access. The components communicate with each other only through the annotation repository. In particular, there is no direct communication between the annotation GUI and the AL component.

4 Experience with Real-World Annotations

We are currently conducting NE annotations for two large-scale information extraction and semantic retrieval projects. Both tasks cover two non-overlapping biomedical subdomains, *viz.* one in the field of hematopoietic stem cell transplantation (immunogenetics), the other in the area of gene regulation. Entity types of interest are, e.g., cytokines and their receptors, antigens, antibodies, immune cells, variation events, chemicals, blood diseases, etc. In this section, we report on our actual experience and findings in annotating entity mentions (drawing mainly on our work in the immunogenetics subdomain) with JANE, with a focus on methodological issues related to active learning.

In the biomedical domain, there is a vast amount of unlabeled material available for almost any topic of interest. The most prominent source is probably PUBMED, a literature database which currently includes over 16 million citations, mostly abstracts, from MEDLINE and other life science sources. We used MESH terms[6] and publication date ranges[7] to select relevant documents from the immunogenetics subdomain. Thus, we retrieved about 200,000 abstracts ($\approx$ 2,000,000 sentences) as our document pool of unlabeled examples for immunogenetics. Through random subsampling, only about 40,000 sentences are considered for AL selection.

For several of our entity annotations, we did both an active learning (AL) annotation and a gold standard (GS) annotation. The latter is performed in the default project mode on 250 abstracts randomly chosen from the entire document pool. We asked different annotators to annotate the same (subset of the) GS to calculate inter-annotator agreement in order to make sure that our annotation guidelines were non-ambiguous. Furthermore, as the annotation proceeds, we regularly train a classifier on the AL annotations and evaluate it against the GS annotations. From this *learning curve*, we can estimate the potential gain of further AL annotation rounds and decide when to stop AL annotation.

4.1 Reduction of Annotation Effort through AL

In real-world AL annotation projects, the amount of cost reduction is hard to estimate properly. We have thus extensively simulated and tested the gain in the reduction of annotation costs of our AL component on available entity annotations of the biomedical domain (GENIA[8] and PENNBIOIE[9]) and the general-

[5] We chose MYSQL, a fast and reliable open source database with native Java driver support

[6] MESH (`http://www.nlm.nih.gov/mesh/`) is the U.S. National Library of Medicine's controlled vocabulary used for indexing PUBMED articles.

[7] Typically, articles published before 1990 are not considered to contain relevant information for molecular biology.

[8] `http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/`

[9] `http://bioie.ldc.upenn.edu/`

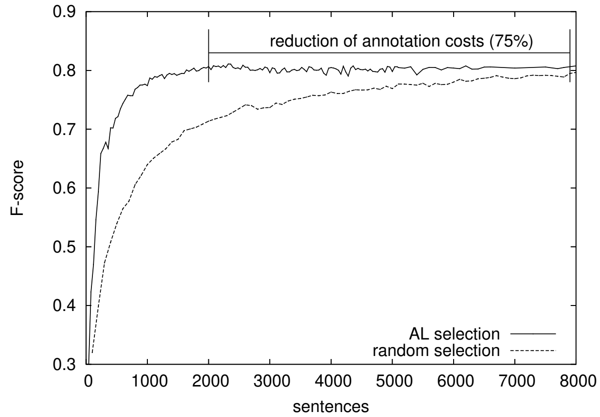

Figure 2: Learning curves for AL and random selection on variation event entity mentions.

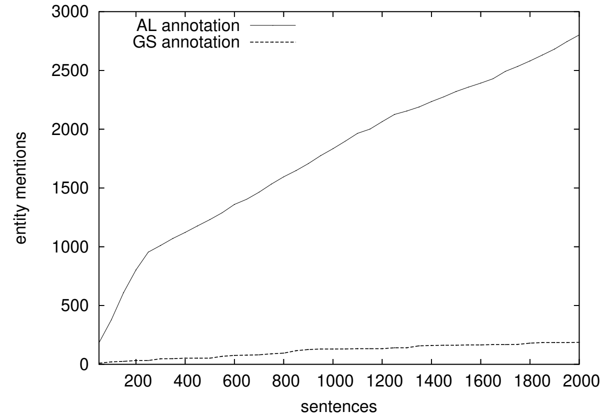

Figure 3: Cumulated entity density on AL and GS annotations of cytokine receptors.

language newspaper domain (English data set of the CoNLL-2003 shared task (Tjong Kim Sang and De Meulder, 2003)). As a metric for annotation costs we here consider the number of sentences to be annotated such that a certain F-score is reached with our NE tagger.[10] We therefore compare the learning curves of AL and random selection. On almost every scenario, we found that AL yields cost savings of about 50%, sometimes even up to 75%.

As an example, we report on our AL simulation on the PENNBIOIE corpus for variation events. These entity mentions include the following six subclasses: type, event, original state, altered state, generic state, and location. The learning curves for AL and random selection are shown in Figure 2. Using random sampling, an F-score of 80% is reached by random selection after $\approx$ 8,000 sentences (200,000 tokens). In contrast, AL selection yields the same F-score after $\approx$ 2,000 sentences (46,000 tokens). This amounts to a reduction of annotation costs on the order of 75%.

Our real-world annotations revealed that AL is especially beneficial when entity mentions are very sparsely distributed in the texts. After an initialization phase needed by AL to take off (which can considerably be accelerated when one carefully selects the sentences of the first AL round, see Section 4.2), AL selects, by and large, only sentences which contain at least one entity mention of the type of interest. In contrast, random selection (or in real annotation projects: sequential annotations of abstracts as in our default project mode), may lead to lots of negative training examples with no entity mentions of interest. When there is no simulation data at hand, the entity density of AL annotations (compared with the respective GS annotation) is a good estimate of the effectiveness of AL.

Figure 3 depicts such a cumulated entity density plot on AL and GS annotations of subtypes of cytokine receptors, really very sparse entity types with one entity mention per PUBMED abstract on the average. The 250 abstracts of the GS annotation only contain 193 cytokine receptor entity mentions. AL annotation of the same number of sentences resulted in 2,800 annotated entity mentions of this type. The entity density in our AL corpus is thus almost 15 times higher than in our GS corpus. Such a dense corpus is certainly much more appropriate for classifier training due to the tremendous increase of positive training instances. We observed comparable effects with other entity types as well, and thus conclude that the sparser entity mentions of a specific type are in texts, the more benefical AL-based annotation actually is.

4.2 Mind the Seed Set

For AL, the sentences to be annotated in the first AL round, the *seed set*, have to be manually selected. As stated above, the proper choice of this set is crucial for efficient AL based annotation. One should definitely refrain from a randomly generated seed set

[10]The named enatity tagger used throughout in this section is based on Conditional Random Fields and similar to the one presented by (Settles, 2004).

– especially, when sparse entity mentions are annotated – because it might take quite a while for AL to take off. If, in the worst case, the seed set contains no entity mentions of interest, AL based annotation resembles (for several rounds in the beginning until incidentally some entity mentions are found) a random selection – which is, as shown in Section 4.1, suboptimal. Figure 4 shows the simulated effect of three different seed sets on variation event annotation (PENNBIOIE). In the tuned seed set, each sentence contains at least one variation entity mention. On this seed, AL performs significantly better than the randomly assembled seed or the seed with no entity mentions at all. Of course, in the long run, the three curves converge. Given this evidence, we stipulate that the sparser an entity type is[11] or the larger the document pool to be selected from is, the later the point of convergence and, thus, the more relevant an effective seed set is.

We developed a useful three-step heuristic to compile effective seed sets without excessive manual work. In the first step, a list is compiled comprised of as many entity mentions (of interest to the current annotation project) as possible. In knowledge- and expert-intensive domains such as molecular biology, this can either be done by consulting a domain expert or by harvesting entity mentions from online resources (such as biological databases).[12] In a second step, the compiled list is matched against each sentence of the document pool. Third, a ranking procedure orders the sentences (in descending order) according to the number of *diverse* matches of entity mentions. This ensures that textual mentions of all items from the list are included in the seed set. Depending on the variety and density of the specific entity types, our seed sets typically consist of 200 to 500 sentences.

4.3 Portability of Corpora

While we are working in the field of immunogenetics, the PENNBIOIE corpus focuses on the subdomain of oncogenetics and provides a sound annotation of these entity mentions (PBVAR).[13] We did a GS annotation on 250 randomly chosen abstracts (≈ 2,000 sentences/65,000 tokens) from our document pool applying PENNBIOIE's annotation guidelines for variation events to the subdomain of immunogenetics (IMVAR-Gold). We then evaluated how well our entity tagger trained on PBVAR would do on this data. Surprisingly, the performance was dramatically low, *viz.* 31.2% F-score.[14]

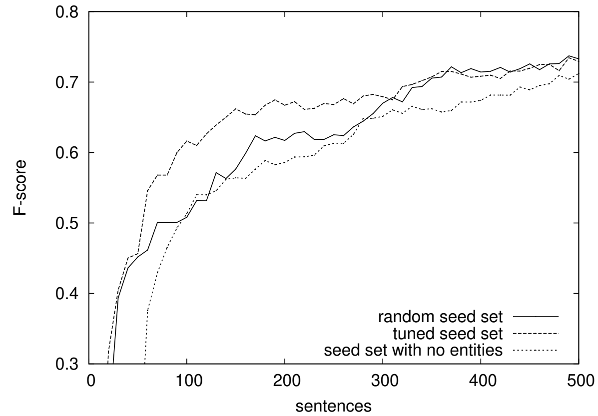

Figure 4: Effect of different seed sets for AL on variation event annotation.

Thus, we did further variation event annotations for the immunogenetics domain with AL: We annotated ≈ 58,000 tokens (IMVAR-AL). We trained our entity tagger on this data and evaluated the tagger on both IMVAR-Gold and PBVAR. Table 1 summarizes the results. We conclude that porting training corpora, even from one related subdomain into another, is only possible to a very limited extent. This may be because current NE taggers (ours, as well) make extensive use of lexical features. However, the results also reveal that annotations made by AL may be more robust when ported to another domain: a tagger trained on IMVAR-AL still yields about 62.5% F-score on PBVAR, whereas training the tagger on the respective GS annotation (IMVAR-Gold), only about half the performance is yielded (35.8%).

[11] Variation events are not as sparse in PENNBIOIE as, e.g., cytokine receptors in our subdomain. Actually, there is a variation entity in almost every second sentence.

[12] In an additional step, some spelling variations of such entity mentions could automatically be generated.

[13] Although oncogenetics and immunogenetics are different subdomains, they share topical overlaps – in particular, with respect to the types of relevant variation entity mentions (such as '*single nucleotide polymorphism*', '*translocation*', '*in-frame deletion*', '*substitution*', etc.). Hence, at least at this level the two subdomains are related.

[14] Note that in a 10-fold cross-validation on PBVAR our entity tagger yielded about 80% F-score.

training data	evaluation data PBVAR	IMVAR-Gold
PBVAR (≈ 200.000 tokens)	≈ 80%	31.2%
IMVAR-AL (58.251 tokens)	62.5%	70.2%
IMVAR-Gold (63.591 tokens)	35.8%	–

Table 1: Corpus portability: PENNBIOIE's variation entity annotations (PBVAR) *vs.* ours for immunogenetics (IMVAR-AL and -Gold).

5 Conclusion and Future Work

We introduced JANE, an annotation environment which supports the whole annotation life-cycle from annotation project compilation to annotation deployment. As one of its major contributions, JANE allows for focused annotation based on active learning, i.e., it automatically presents sentences for annotation which are of most use for classifier training.

We have shown that porting annotated training corpora, even from one *sub*domain to another and thus related to a good extent, may severely degrade classifier performance. Thus, generating new annotation data will increasingly become important, especially under the prospect that there are more and more real-world information extraction projects for different (sub)domains and languages. We have shown that focused, i.e., AL-driven, annotation is a reasonable choice to significantly reduce the effort needed to create such annotations – up to 75% in a realistic setting. Furthermore, we have highlighted the positive effects of a high-quality seed set for AL and outlined a general heuristic for its compilation.

At the moment, the AL component may be used for most kinds of segmentation problems (e.g. POS tagging, text chunking, entity recognition). Future work will focus on the extension of the AL component for relation encoding as required for coreferences or role and propositional information.

Acknowledgements

We thank Alexander Klaue for implementing the GUIs. This research was funded by the EC within the BOOTStrep project (FP6-028099), and by the German Ministry of Education and Research within the StemNet project (01DS001A to 1C).

References

Lynn Carlson, Daniel Marcu, and Mary E. Okurowski. 2003. Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. In J. van Kuppevelt and R. Smith, editors, *Current Directions in Discourse and Dialogue*, pp. 85–112. Kluwer.

Sean Engelson and Ido Dagan. 1996. Minimizing manual annotation cost in supervised training from corpora. In *Proc. of ACL 1996*, pp. 319–326.

B. Hachey, B. Alex, and M. Becker. 2005. Investigating the effects of selective sampling on the annotation task. In *Proc. of CoNLL-2005*, pp. 144–151.

Rebecca Hwa. 2000. Sample selection for statistical grammar induction. In *Proc. of EMNLP/VLC-2000*, pp. 45–52.

Jin-Dong Kim and Jun'ichi Tsujii. 2006. Corpora and their annotation. In S. Ananiadou and J. McNaught, editors, *Text Mining for Biology and Biomedicine*, pp. 179–211. Artech.

M. Marcus, B. Santorini, and M. A. Marcinkiewicz. 1993. Building a large annotated corpus of English: The PENN TREEBANK. *Computational Linguistics*, 19(2):313–330.

C. Müller and M. Strube. 2003. Multi-level annotation in MMAX. In *Proc. of the 4th SIGdial Workshop on Discourse and Dialogue*, pp. 198–207.

Grace Ngai and David Yarowsky. 2000. Rule writing or annotation: Cost-efficient resource usage for base noun phrase chunking. In *Proc. of ACL 2000*, pp. 117–125.

Tomoko Ohta, Yuka Tateisi, and Jin-Dong Kim. 2002. The GENIA corpus: An annotated research abstract corpus in molecular biology domain. In *Proc. of HLT 2002*, pp. 82–86.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

David Pierce and Claire Cardie. 2001. Limitations of co-training for natural language learning from large datasets. In *Proc. of EMNLP 2001*, pp. 1–9.

James Pustejovsky, Patrick Hanks, Roser Saurí, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, and Marcia Lazo. 2003. The TIMEBANK corpus. In *Proc. of the Corpus Linguistics 2003 Conference*, pp. 647–656.

Burr Settles. 2004. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proc. of JNLPBA 2004*, pp. 107–110.

H. Sebastian Seung, Manfred Opper, and Haim Sompolinsky. 1992. Query by committee. In *Proc. of COLT 1992*, pp. 287–294.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CONLL-2003 shared task: Language-independent named entity recognition. In *Proc. of CoNLL 2003*, pp. 142–147.

Katrin Tomanek, Joachim Wermter, and Udo Hahn. 2007. An approach to downsizing annotation costs and maintaining corpus reusability. In *Proc of EMNLP-CoNLL 2007*.

Kees van Deemter and Rodger Kibble. 2000. On coreferring: Coreference in MUC and related annotation schemes. *Computational Linguistics*, 26(4):629–637.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2/3):165–210.