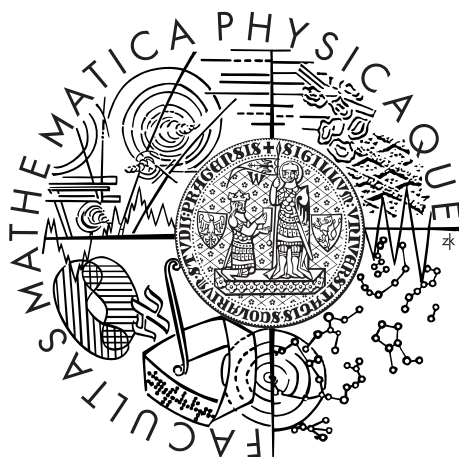


Charles University in Prague  
Faculty of Mathematics and Physics

## MASTER THESIS



Benno Weck

# Assessing the Impact of Manual Corrections in the Groningen Meaning Bank

Institute of Formal and Applied Linguistics

Supervisor of the master thesis: Markéta Lopatková

Study programme: Informatika

Specialization: Matematická lingvistika  
(EM LCT)

Prague 2015

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular the fact that the Charles University in Prague has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 paragraph 1 of the Copyright Act.

In ..... date .....

signature of the author

**Název práce:** Posouzení účinku manuálních oprav na Groningen Meaning Bank

**Autor:** Benno Weck

**Katedra:** Ústav formální a aplikované lingvistiky

Vedoucí diplomové práce: RNDr. Markéta Lopatková, Ph.D. - Ústav formální a aplikované lingvistiky

**Abstrakt:**

Projekt Groningen Meaning Bank (GMB) vytváří korpus s bohatou syntaktickou a sémantickou anotací. Anotace v GMB jsou generovány poloautomaticky na základě dvou zdrojů: (i) Vstupní anotace ze sady standardních nástrojů pro zpracování přirozeného jazyka (NLP) (ii) Opravy/vylepšení od lidských anotátorů.

Například na úrovni anotace slovních druhů existuje 18 000 takových oprav, nazývaných *Bits of Wisdom* (BOWs). V této práci zkoumáme možnosti zlepšení technik NLP pomocí zapojení těchto informací. V experimentech používáme BOWs pro přetrénování analyzátoru slovních druhů. Zjistili jsme, že analyzátor může být vylepšen tak, aby opravil až 70% nalezených chyb v testovacích datech. Tento zlepšený analyzátor navíc napomáhá ke zlepšení výkonu parseru. Nejspolehlivější cestou se ukázalo být preferování vět s vysokou mírou potvrzených analýz po přetrénování.

V experimentu se simulovaným aktivním učením používajícím Query-by-Uncertainty (QBU) a Query-by-Committee (QBC) jsme ukázali, že selektivní vzorkování vět pro přetrénování dává lepší výsledky a vyžaduje méně dat než použití náhodného výběru.

V doplňkové pilotní studii jsme zjistili, že standardní analyzátor slovních druhů trénovaný modelem maximální entropie může být rozšířen použitím známých analýz ke zlepšení svých rozhodnutí na celé sekvenci bez přetrénování modelu.

**Klíčová slova:** korpus, slovní druhy, anotace, opravy, NLP

**Title:** Assessing the Impact of Manual Corrections in the Groningen Meaning Bank

**Author:** Benno Weck

**Department:** Institute of Formal and Applied Linguistics

**Supervisor:** RNDr. Markéta Lopatková, Ph.D. - Institute of Formal and Applied Linguistics

**Abstract:**

The Groningen Meaning Bank (GMB) project develops a corpus with rich syntactic and semantic annotations. Annotations in GMB are generated semi-automatically and stem from two sources: (i) Initial annotations from a set of standard NLP tools, (ii) Corrections/refinements by human annotators.

For example, on the part-of-speech level of annotation there are currently 18,000 of those corrections, so called *Bits of Wisdom* (*BOWs*). For applying this information to boost the NLP processing we experiment how to use the *BOWs* in retraining the part-of-speech tagger and found that it can be improved to correct up to 70% of identified errors within held-out data. Moreover an improved tagger helps to raise the performance of the parser. Preferring sentences with a high rate of verified tags in retraining has proven to be the most reliable way.

With a simulated active learning experiment using Query-by-Uncertainty (QBU) and Query-by-Committee (QBC) we proved that selectively sampling sentences for retraining yields better results with less data needed than random selection.

In an additional pilot study we found that a standard maximum-entropy part-of-speech tagger can be augmented so that it uses already known tags to enhance its tagging decisions on an entire sequence without retraining a new model first.

**Keywords:** corpus, part-of-speech, annotation, correction, NLP

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Background</b>	<b>7</b>
2.1	The Groningen Meaning Bank . . . . .	7
2.2	The statistical tools: The C&C tools . . . . .	9
2.3	Part-of-speech Tagging: An example of Sequence Tagging . . . . .	10
<b>3</b>	<b>Related Work</b>	<b>14</b>
3.1	Crowdsourcing: Non-experts in annotation . . . . .	14
3.2	Optimal Sampling: Active Learning . . . . .	15
3.3	Error detection in annotated corpora . . . . .	17
<b>4</b>	<b>Method</b>	<b>18</b>
4.1	Data Sets . . . . .	20
4.1.1	Training Data Set . . . . .	20
4.1.2	Creating Silver Standard and Selecting Gold Standard Test Data . . . . .	20
4.2	Sampling strategies for selecting additional training data . . . . .	22
4.2.1	Random Sampling . . . . .	23
4.2.2	Longest sentence first . . . . .	23
4.2.3	Cautious sampling . . . . .	23
4.3	The Active Learning approach . . . . .	24
4.3.1	Query by Uncertainty . . . . .	25
4.3.2	Query by Committee . . . . .	26
4.4	Impact on higher levels of annotation — (Syntactic level) . . . . .	26
<b>5</b>	<b>Results and Discussion</b>	<b>28</b>
5.1	Random, Longest Sentence First, & Cautious Sampling . . . . .	28
5.2	The Active Learning approach . . . . .	31

5.3	Impact on higher levels of annotation — (Syntactic level) . . . . .	34
5.4	General Discussion . . . . .	36
<b>6</b>	<b>Pilot Study: Building a smarter tagger</b>	<b>38</b>
6.1	Method . . . . .	38
6.2	Results . . . . .	40
6.3	Discussion . . . . .	42
<b>7</b>	<b>Conclusion</b>	<b>45</b>
7.1	Findings . . . . .	45
7.2	Advice . . . . .	46
7.3	Future Work . . . . .	46
	<b>Bibliography</b>	<b>48</b>
	<b>List of abbreviations</b>	<b>54</b>
	<b>Appendices</b>	<b>55</b>

# 1. Introduction

To a lot of tasks in Natural Language Processing (NLP), a corpus – a large collection of (annotated) text – is indispensable. Since corpora can be seen as a representation of a language, they provide the means of studying language qualitatively and quantitatively. They are the basis for an exploratory data analysis or for corpus studies that can give insight to specific linguistic phenomena, help to spot differences between different languages or can provide a way to prove or disprove a theory. Moreover, annotated corpora, i.e. labeled data, are needed for supervised learning. They are widely used as training and testing material for statistical and machine-learning-based systems, that address all kinds of classic NLP problems. For example, probabilistic methods for named entity (NE) recognition or other information extraction problems heavily rely on annotated data for training purposes [50].

Assembling an annotated corpus, however, can be a challenging and cost-intensive endeavor. While collecting raw text data in large quantities is becoming easier as more texts are available in digital form [4], annotating it usually still consumes the largest portion of time and money. Traditionally, the biggest item in terms of money and time is human labour for manually labeling text data. Cost constraints, however, are usually the limiting factor when assembling a corpus and render manual annotation for large corpora infeasible. Fortunately, there are automatic tools (pre-trained on already existing corpora), which provide fairly accurate annotations with little to no additional cost, that can help greatly in the annotation procedure. For example in the setting of part-of-speech (POS) tagging, taggers are reported to achieve an accuracy of above 97% [36]. Despite very good performance of statistical tools on some annotation tasks, they are not perfect and sometimes produce erroneous annotation, for example on unknown words. In order to create a high-quality corpus with reliable annotation, those errors require correction. This correction is typically again provided by human annotators. It is aimed for high quality, while their manual effort is kept to a minimum. This is why combining automatic pre-annotation and correction seems to have obvious advantages over full annotation from scratch since it greatly reduces the amount of manual work needed in the process of corpus creation. In our research we are interested in investigating the effect of those (manual) corrections of automatically annotated text. For this we work with the Groningen Meaning Bank (GMB) corpus [5].

The GMB project compiles and maintains a corpus of texts with manifold annotation, including deep semantic representations of the texts. In the process of building this resource, statistical state-of-the-art NLP tools were used to provide the first step of annotation. The annotations are continuously refined through manual corrections provided by experts and non-experts. Annotation can also be augmented or replaced by external annotation from trusted sources like other corpora or special NLP tools. Within the GMB project these corrections and external knowledge are referred to as Bits of Wisdom (BOWs). This approach guarantees a steady improvement of the quality of annotation as long as (more) human effort is invested. However it is not clear to date how efficient this improvement is.

Consider the following example sentences:

- (1) a. Exercise every day!
- b. Exercise regularly to be fit!

In the sentence given in (1a), the word ‘*Exercise*’ was incorrectly classified as a noun. This error was corrected to the accurate verb classification. We can assume that the tagger is also mistaken on similar cases like the one given in (1b) or that it also misclassified other verbs in the imperative form as a noun. This error still persists after the correction on the first sentence since this only corrects the misclassification locally. This means that the correction has no influence on the tagging in the rest of the corpus. Identifying all similar cases and correcting them would be a laborious task. Besides, if for example more data gets added to the corpus and is also automatically annotated, a newly added similar case would still be misclassified. This is certainly an unwanted practice since it misses any learning effect. Rather it is desirable that the already provided information (i.e. the correction in the first example) helps to solve errors in all similar cases (e.g. the second sentence). Moreover, correcting similar errors repeatedly gives gradually less new information from correction to correction. The need of human involvement should be minimized, while the impact of manual corrections is maximized.

In order to avoid further misclassifications the classifier has to learn the corrections that are provided. In general retraining the statistical tool, employed for providing the annotation, with the newly gained information/data is the only available method to implement this demand. While we have to make sure that the corrections are actually learned by the classifier, retraining raises at the same time the possibility and problem of introducing new errors. If we recall the examples from above, we do not want that learning the interpretation of ‘*Exercise*’ as a verb causes other occurrences of this word to be misclassified as a verb. For example, in the sentences given in (2) the classification of ‘*Exercise*’ as a noun should not change.

- (2) Exercise is the key to good health.

The central goal of this thesis is to assess the possible impact and effect of manual corrections in the GMB. We want to address this objective by investigating and answering a number of questions:

- How can the models of the statistical annotation tools be retrained effectively?
- How can the effect of a single BOW be maximized?
- How can the supervision of the retraining process be minimized?
- What can be learned from the existing corrections for future annotation effort?

Answering these questions will help us find ways for exploiting the existing corrections and effectively improving the annotation in the corpus. This will bring us one step closer to building a high quality gold-standard resource that can be of great use to the research community. Additionally, with repeated

iterative retraining it might be possible to build a robust automatic annotation system, that reduces the need for manual correction in the future. With our experiments we also seek to gain knowledge useful for the further development of the GMB corpus. At the moment BOWs are provided in an unguided fashion, i.e. annotators are not specifically guided in their way where to look for possible errors or where verification of annotation might be needed. This has a number of implications, of which we are interested in the issues related to the efficiency of the annotators. Most prominently it leads to the consequences that the provided corrections can be of low value to the whole corpus or the provided corrections are redundant. These two problems are immediately connected with each other. Correcting a sentence might not provide much new information to the tagger when it is added to the training data. The reasons for that might be that the corrected error is infrequent in the corpus or only one word out of a long sequence of reliably tagged tokens is corrected. In the case of only one corrected word out of the entire sequence, only this word really contributes to the training. Many similar corrections are likely to have only little added value in training. This is problematic because we want maximum impact from the corrections since they are costly manual effort.

As an extra to this main research focusing on retraining, we will investigate another approach to make (additional) use of the corrections provided in our data. We execute a small-scale pilot study on how corrections can be made available to the tagger already at tagging time without retraining a new model first. We consider it an unnecessary restraint that the tagger cannot use the already known tags in a sequence and is thus blind to existing information. We hypothesize that, if this information is used by the tagger, the tagging of other tokens is positively influenced through reduced uncertainty in the sequence. Since this experimental design explicitly avoids retraining, we separated it from the other experiments and discuss it in an extra chapter. Its results, however, will add to the outcomes of the rest of our research.

There are many different layers of annotation with existing corrections in the GMB, but for this thesis we will concentrate on the POS level of annotation. The choice of this level of annotation is motivated by two reasons. First, this level has the highest amount of correction in comparison to the other levels of annotation. Because of the high distribution of corrections on this layer, it is not only the most interesting starting point, but it also will probably give the most reliable results. Secondly, POS tagging takes place in the beginning of the processing pipeline (only preceded by text segmentation) and higher levels need it as input. The entire pipeline will profit from improvements made in early stages. Additionally, since text segmentation operates on almost 100% accuracy, hardly any errors are propagated from other tools earlier in the processing. This means that no errors originating from other tools influence the results of our experiments.

The thesis will be structured as follows. First, we want to introduce the GMB to outline the context in which this work was executed, as well as provide the reader with relevant background information about POS tagging in general and the statistical tools being used. After this an overview of related work relevant to our research is given. The main experimental setup of this research is presented in chapter 4. The results are discussed in chapter 5. In chapter 6 we propose an enhancement to the general POS tagger design in a proof-of-concept

study. Finally we will conclude by summing up the results and answering the research questions. Additionally, we draw conclusions for the GMB project and give pointers for possible future work. We give a list with short explanations for the POS tags used throughout the thesis in the appendix.

## 2. Background

In this second chapter we want to give background information on three main elements the reader should be familiar with in order to fully understand the implications of our work: (i) the corpus that we work with (including information about the manual correction/annotation available in the corpus), (ii) the statistical tools that provide annotation in the corpus and (iii) automatic part-of-speech annotation.

### 2.1 The Groningen Meaning Bank

The Groningen Meaning Bank (GMB) [5, 10] is a publicly available corpus that provides texts with syntactic and deep semantic annotation. The declared goal of the GMB project is to create a gold standard corpus of meaning representations. To achieve this goal, automatic annotation from state-of-the-art NLP tools is combined with corrections and adjustments from both expert and non-expert annotators. For the semantic representation of the texts a variant of the Discourse Representation Theory [31] was chosen. This theory employs Discourse Representation Structures (DRSs), that can capture several linguistic phenomena, as its basic meaning carrying units.

The fairly complex processing pipeline in the GMB provides a number of annotations on multiple layers. The tools that make up the pipeline will be discussed shortly in the next section (2.2). Processing takes place on the following levels:

1. sentence boundary detection/tokenization
2. POS tagging
3. NE tagging
4. supertagging (assigning the grammatical (CCG) categories)
5. parsing (syntactic analysis)
6. boxing (semantic analysis)

The GMB corpus consists of 100 parts, and the latest release (version 2.2.0) contains 10,000 documents with 1,354,149 tokens. Since the GMB project is an ongoing effort, more texts are added regularly. The current development version includes an additional 103 documents. Moreover the project currently maintains a backlog of roughly 63,000 documents with more than 32M tokens that are placed at its disposal and ready to be checked but not accepted into the corpus yet. The development version is accessible through the wiki-like interface,<sup>1</sup> shown in Figure 2.1, that allows easy editing of tokenization or annotation [6].

It is noteworthy that the GMB includes not only newswire text, a usual prevailing genre of a corpus, but also jokes, fables and country descriptions. To assure that the genres are distributed evenly in the corpus, the documents of the respective subcorpora are spread across the parts. The GMB orientates itself

---

<sup>1</sup> <http://gmb.let.rug.nl/explorer>



Figure 2.1: The Explorer of the GMB in an Internet browser showing a document with POS annotation.

on the CCGbank [29] for the representation of syntactic information and uses a combinatory categorial grammar (CCG). CCG is a lexicalized theory of grammar [59]. This means that most work is done on the word level (in the lexicon) and only few grammar rules are needed. This makes it very easy to use with manual corrections, since changes are done on the lexical category of a token rather than by annotation of complex syntactic structures. Consequently also the tagset for POS annotation is geared towards the tagset of the CCGbank with minor changes. The tagset of the CCGbank is an extension of the classic Penn Treebank tagset [37].

The primary language of the GMB project is English, but currently an effort is made to integrate more languages to create a parallel meaning bank [10]. Similar to parallel treebanks a parallel meaning bank could help improve machine translation algorithms.

A fundamental concept in the design and the development of the GMB is the notion of *Bits of Wisdom (BOWs)*. All changes and corrections made to the data in the corpus at any stage of the processing are subsumed by this concept. The BOWs are designed to be information preserving, i.e. any change can be reverted. As a consequence they are traceable and easy to administer, and this is especially useful as there are a number of different sources. The two main sources for BOWs are expert annotators and non-expert annotators. Experts can correct through an online interface and at any level of annotation. The ‘wisdom’ of the non-experts comes from the game with a purpose (GWAP) *Wordrobe*<sup>2</sup>, that was created specifically for this purpose [66]. In this game players solve linguistic problems (e.g. choosing the right grammatical category for a word) in a playful way. In contrast to the experts not every suggestion by a player is directly adopted as a BOW, but only those that are supported by a larger number of players. BOWs can also come from external tools like for example a word-sense-disambiguation system. As BOWs are applied in every step in the pipeline, they do not only correct the final result of annotation, but the corrected data are also available to higher levels of processing. If two or more BOWs contradict each other, a judging

<sup>2</sup> <http://wordrobe.org/>

component is employed. So far, judging follows a simple strategy by preferring more recent BOWs and ranking expert annotators over every other source. More sophisticated ways of resolving conflicting BOWs might follow in the future.

The total number of active POS BOWs provided by a human annotator (in the current development version) is 18,006. There are 7,143 documents that contain at least one POS BOW. From Figure 2.2 we can see that the 10 first sections in the GMB contain a slightly higher amount of POS BOWs than the rest. In an overall view the BOWs are distributed reasonably evenly. As we can see from the chart in Figure 2.3 the POS level represent 12% of all BOWs. The NE level has a significantly higher amount of BOWs which is partially due to the fact that NEs often span multiple tokens and thus more tokens get annotated with a single correction.

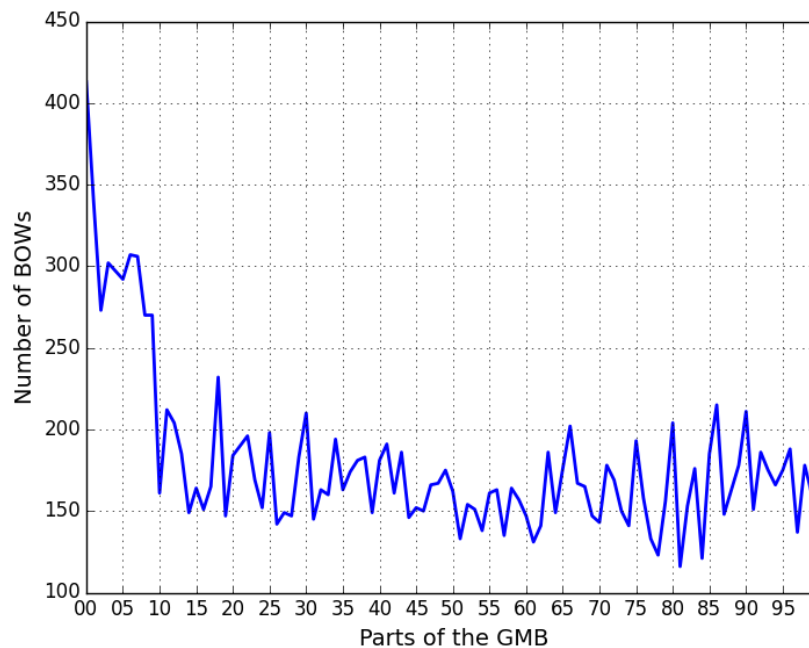


Figure 2.2: Distribution of BOWs on the POS level over the 100 parts of the GMB

## 2.2 The statistical tools: The C&C tools

The automatic text processing for the GMB corpus is done by a number of statistical tools. Most of the tools in the pipeline are part of the *C&C* tools<sup>3</sup> [20]. The pipeline handles one document at a time in multiple different steps that provide annotation in a cascading fashion.

Text segmentation (sentence splitting & tokenization) is the first step in the processing and is done by the statistical tool *elephant* [22]. The centerpiece of the *C&C* pipeline is a wide-coverage CCG parser (from the *C&C* tools) [14]. The annotation required by the parser as input features is provided by a number of maximum entropy (ME) taggers. These taggers include a POS tagger, an NE

<sup>3</sup> <http://svn.ask.it.usyd.edu.au/trac/candc>

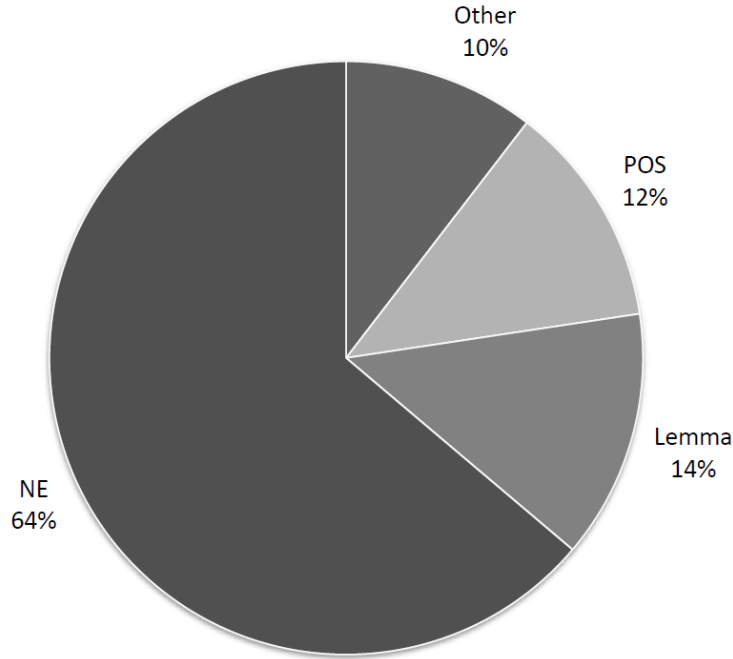


Figure 2.3: Annotation levels with the biggest portion of BOWs

recognizer, and a CCG supertagger [20, 19, 18]. The ME framework for tagging is also shortly covered in the following section (2.3). All of these taggers are part of the *C&C* toolchain. The pipeline is completed by *boxer* [9]. This last component generates the semantic representations in the form of DRSs.

The Figure 2.4 shows an overview how the tools interact to produce the DRS representation output from a document as input.

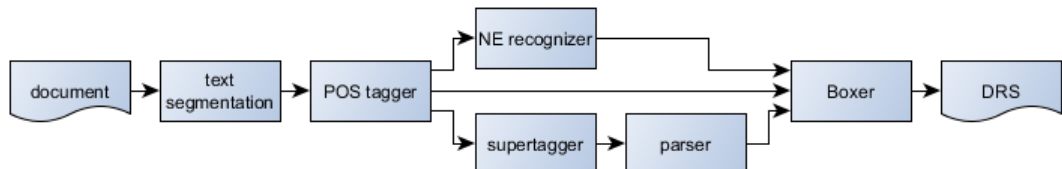


Figure 2.4: Schematic representation of the processing pipeline in the GMB

## 2.3 Part-of-speech Tagging: An example of Sequence Tagging

Part-of-speech (POS) tags are descriptive labels which are assigned to a token, i.e. a word or a punctuation mark, in a text. As the name already indicates, these tags describe the part-of-speech (noun, verb, adjective, ...) of a token. But this is only a vague definition. The linguistic information contained in the tag differs with tagset and language, and can include morphological and lexico-semantic properties as well. For example this can include information about number, grammatical gender or voice. Another name, proposed by van Halteren [63] as

more ‘adequate’, is *morphosyntactic tags*. POS tagging is a form of sequence tagging and the process of tagging involves disambiguating, determining a basic form, lemma, and (automatically) assigning the tags to each token in a sequence (usually a sentence). When considering the form of the words ‘*study*’ and ‘*work*’ in the example (3), both could be a verb or a noun. To be able to interpret the sentence the two words have to be disambiguated. In this case ‘*study*’ should be tagged as a noun and ‘*work*’ as a verb.

(3) I work on a study of language.

POS tagging is a prerequisite for a number of other NLP tasks, as they use this preprocessing step as an input for more complex analysis (e.g. parsing). POS tagging adds an extra layer of information to the text data at hand and is one step for the interpretation of the text. For example in machine translation, adding POS information to a phrase in the source language can help disambiguate it and thus facilitate finding the correct translation in the target language [62].

A comprehensive overview of the history of POS tagging in the past decades is given by Voutilainen [67] and van Halteren [63]. Over the years POS taggers have moved from linguistically motivated, rule-based approaches to data-driven machine learning (ML) and statistical techniques. Despite their expressive power and their property of being able to tackle sophisticated linguistic knowledge, hand-written rules are more limited and not easily transferable to other languages, and domains/tagsets. The first statistical systems were able to easily outperform systems with hand-written rules in terms of accuracy. Data-driven systems, in general, learn a language model from training data to disambiguate words. A basic, but effective, way of doing this is storing frequency information of short word-tag-pair sequences (n-grams). This information about the frequency of a tag for a given word in an n-gram context can then be easily applied in tagging. For example, if a word that is noun-verb ambiguous is preceded by an unambiguous determiner (e.g. ‘*the*’), the noun reading is chosen as this is more frequent and thus more probable.

For morphologically rich languages this method has to be augmented since many word forms are especially sparse in these languages. It is infeasible to list all possible word forms with frequency estimates since there are too many possibilities and estimating the frequency from data is challenging. A solution is to use the lemma (base form) of words instead.

A common paradigm in POS tagging is the Hidden Markov Model (HMM). The taggers are popular for their processing speed and accuracy. An HMM does not encode the information about the tags (probability) explicitly. The underlying assumption of this model is that the sequence of words that needs to be tagged was ‘generated’ from a (hidden) sequence of tags. The probability of a tag in this sequence depends on a fixed history (e.g. two preceding tags in case of a trigram model). The probability of a word to be ‘generated’ depends on the POS tag. The algorithm used for tagging essentially tries to recover the (most likely) sequence of tags that ‘generated’ the string in question. Similar to all statistical methods these parameters are usually estimated from annotated text. However, an advantage of HMM-based taggers is that they can be learned from untagged text only with the help of a lexicon (which gives the allowed tags for a certain word). For this, an HMM model is initialized and the model parameters are

refined iteratively by reducing the ambiguity measured on a training corpus with the help of the Baum-Welch algorithm [7]. HMM taggers that are trained on tagged data, however, usually have higher accuracy than those learned from raw text [51, 67].

An approach that includes the notion of rules in data-driven learning is transformation-based tagging. In this approach a set of ordered rules is learned in an error-driven manner from labeled training data. Besides a tagged training corpus a set of predefined local rule templates is needed from which rules can be learned. In the learning phase the training corpus is tagged according to some (initial) rules and compared to the true annotation. A new rule is created from a template and adopted to refine the so far defined rules in order to minimize misclassifications if it brings the best correction rate. In tagging the rules are applied in the order that was learned [67].

A widely used state-of-the-art method for POS tagging is the maximum entropy (ME) framework (sometimes also MaxEnt for short). A tagger algorithm using this framework was put forward by Ratnaparkhi [42] and his implementation MXPOST reaches state-of-the-art accuracy (96.6%) on a test set for English. The ME model gives the conditional probability of a tag in a context. The contextual, probabilistic information comes from a set of binary contextual features (predicates). These contextual predicates usually include information about the surrounding tags and tokens. The model keeps a weight for all of the predefined features and in training, the weights for features are learned. The fundamental concept of ME modeling is, that out of a set of potential models always the most uniform model, which satisfies a set of constraints, is chosen. The model with the highest entropy is the most uniform [17]. Hence the name maximum entropy. The model is constrained by the expected value of each feature. The (approximated) expected value according to the model should match the expected value empirically observed in the training data. The tagger used in this work (mentioned in section 2.2) is an ME tagger. The performance of an ME tagger can further be refined by smoothing the model. Smoothing can help to avoid overfitting by relaxing the model constraints on low-frequency features [17].

The performance of a POS tagger is most commonly measured by its accuracy (sometimes also expressed as error rate). The correctness of the tagger is essential, whereas processing speed and memory requirements are nowadays negligible factors. To evaluate the tagger, it is run on a corpus with trusted annotation, a gold standard corpus, which is typically hand-annotated. By comparing the tagger’s output to the gold standard data, the accuracy (the ratio of correctly tagged tokens to all tokens) can easily be computed. Related measures, taken from information retrieval, are precision and recall. These measures allow for a more fine-grained analysis of results. For example, one can calculate precision and recall for a specific tag to see whether too many false positives are produced.

The informative value of the tagset used with the tagger (in matters of the ambiguity present in the input) influences the avail of the tagger output. It is ‘easier’ to tag a text with a tagset only containing a few tags since less choices need to be made, but the information content might be bigger if a larger and more delicate tagset is used. However this concept is more difficult to measure, since no clear evaluation metric is defined.

Tagsets used in POS tagging do not only differ for different languages but also within one language usually multiple tagsets with varying specifications, granularity and features are in use. The differences in the tagsets reflect needs of different target applications or specific underlying linguistic theories. There exist a few popular tagsets which are used for a number of applications and thus allow for easy comparability of different systems. The Penn Treebank tagset [37] is probably the most prevalent for English. With the diversity of tagset comes the problem that information is often trapped in one tagset specification. Zeman [70] presents a general means to convert linguistic information from one tagset to any other tagset. A set of morphosyntactic features (the *Interset*) that is designed to be universal and capture all information possibly present in a POS tag is the medium to translate between tagsets. Once it is defined how a tagset can be converted into the *Interset* (and how a tag can be found from the *Interset*) it can be converted to any other tagset that is convertible to *Interset*. Another notable effort to make linguistic annotation more generally accessible is the definition of a universal tagset for POS tagging. A universal tagset that is aimed to be applicable across languages is given by Petrov et al. [41]. In contrast to the *Interset*, this tagset is specifically meant to be applied in tagging. As a consequence converting a more specific tagset to this universal tagset implies that information is lost.

The input to a POS tagger is a properly segmented (tokenized) text, the output is a sequence of triples consisting of a token, its disambiguated lemmas and its POS in the given utterance. Two key constituents are shared by most classic POS tagging architectures: (i) ambiguity look-up and (ii) disambiguation. First, all possible POS tags for the word in question have to be listed. This can be done by a simple lexicon look-up or by sophisticated guessing. Secondly, the potential tags have to be ranked or a single one has to be chosen as the correct tag. This analysis builds on information about the word itself (e.g. frequency of the word appearing in a certain part-of-speech) and contextual information. The latter generally includes the surrounding words and their POS tags. Often also the goal of maximizing the overall probability of a tag sequence rather than just the probability of a single tag influences the choice of a specific tag. As a lexicon can never be exhaustive, taggers also have to deal with unseen words. Contextual information or morphological analysis (e.g. analyzing the affix of a word) are often used as indicators of the correct tag.

The consensus view seems to be that POS tagging is considered a solved problem. This is based on the fact that current state-of-the-art taggers for English manage to surpass up to 97% accuracy [36]. Despite of the fact that POS tagging is considered a solved problem, there is still research actively involved with developing it further and uncovering its limitations. Giesbrecht and Evert [25] challenge the consensus that POS tagging is a solved problem. On the basis of experiments they show that only if training and test set are similar a very high accuracy can be achieved. Manning [36] gives a critical analysis on what prevents POS tagging of reaching 100% accuracy. He finds that there is a number of different groups of errors, which can not all be addressed by improving the semi-supervised learning component of the tagger and thus call for different resolutions. For example, one of the main sources of errors are inconsistently or wrongly tagged data in popular gold-standard corpora. This implies that a tagger cannot reach 100% on a test set containing errors.

### 3. Related Work

The discipline of corpus linguistics has a long history in collecting and using text for linguistic research. Nowadays there is a large number of corpora available in multiple languages, for both written text and transcribed speech. In the literature, we find various examples of experiences made by scholars in projects involved with the construction of corpora. It seems that there is broad consensus in best practices how to design the construction of a corpus. Those practices address issues like copyright, balancing the data, data acquisition and preparation [32, 38]. Basic procedures of annotation and preprocessing that are commonly used for preparing data for linguistic research are also often used in the creation of a corpus.

This chapter focuses on three major areas of research: first we want to showcase a different approach used in corpus annotation in contrast to the ‘traditional’ way of manual annotation by an expert. Crowdsourcing, or annotation by non-experts in general, is one way to reduce annotation costs. Then we discuss Active Learning (AL) as a way of accelerating corpus annotation by bootstrapping it in more depth, as this is of particular interest to our research. Finally, by discussing research on error detection and correction in annotated corpora we hope to shed light on the importance of correcting errors in an annotated corpus.

#### 3.1 Crowdsourcing: Non-experts in annotation

To tackle the problem of costly expert annotation, the community has come up with different ideas to delegate the work to non-experts. Employing non-expert annotators has proven to be a cost-effective and still reliable way to gather annotation. The idea behind crowdsourcing is that a final annotation decision is the product of a number of single decisions of several amateur agents. Thus the ‘wisdom’ of the crowd is comparable to the ‘wisdom’ of a few experts.

Wang et al. [68] and Sabou et al. [48] compare different ways of crowdsourcing against the background of their applicability in NLP and try to give best practice guidelines. The three main approaches to involve non-experts in data preparation tasks are crowdsourcing with paid workers, games with a purpose (GWAPs) and altruistic work by volunteers. The biggest difference is that only in the first case participants are rewarded financially. In this case annotation cannot only be gathered cheaper but also faster, by distributing the annotation tasks via the Web to many subjects that are paid small amounts of money for small tasks. This type of crowdsourcing has been used with success in linguistic annotation tasks like word sense disambiguation [57].

In a GWAP participants play a game to generate or validate data [13]. Players are attracted by an entertaining game design. An example of such a game is *Phrase Detectives* [12]. In this game players help in annotating anaphoric information.

Systems that are open for collaborative editing, such as *Wikipedia*<sup>1</sup>, have proven to attract volunteers that are willing to generate or validate content with-

---

<sup>1</sup> <https://www.wikipedia.org/>

out special reward.

All three approaches come with a certain setup effort. While data preprocessing has to take place in every annotation setting it is of particular importance when employing non-experts to ensure a high usability. Better usability will attract more subjects and positively influence annotation quality. For employing paid workers there already exist a number of platforms with a permanent worker base. For example the platform Amazon Mechanical Turk<sup>2</sup> was successfully used in linguistic research [52]. GWAPs have the disadvantage that they often have to be designed and implemented from scratch and have to establish a user base. Once they are in operation only little extra costs incur.

Within the GMB project two of the three ways (GWAP & effort from volunteers) of facilitating annotation have been used for different kinds of annotation tasks [10]. As described above (in section 2.1), a volunteer (usually an expert) is free to edit any level of annotation with the help of an online interface. In the GWAP platform *Wordrobe*<sup>3</sup> players solve a number of tasks including POS tagging, NE tagging, co-reference resolution and word sense disambiguation by playing different games [66].

## 3.2 Optimal Sampling: Active Learning

An increasingly popular form of bootstrapping an annotated corpus is Active Learning (AL). It is sometimes also called *optimal sampling* or *selective sampling* [45]. AL aims to reduce the amount of training data which need to be labeled (additionally) by selecting those samples out of a pool of unannotated data, which contribute the most. Starting from an initial training set a learner selects a number of instances out of a set of unlabeled data and queries an oracle for their annotation. This oracle is typically a human annotator. The newly labeled data are then added to the training set and the algorithm proceeds in a iterative fashion until a stopping criterion is met. In this way it can help to speed up annotation and to minimize human involvement. AL was successfully applied in a number of NLP tasks, including POS tagging [45], NE tagging [61, 26, 8] and parsing [40]. AL is not only used in the setting of NLP, but also in other ML related tasks. An encyclopedic overview over the research on AL is given by Settles [53]. Most research on AL in NLP mainly concentrates on finding the best selection algorithm for a specific task or establishing a benchmark for expected reduction in training needed. The results are grounded on simulated (the oracle is simulated by using pre-labeled data) or real life experiments. A common way to estimate the training utility of a sample is to measure the uncertainty of a classifier (or a set of classifiers). An extensive survey of many different selection strategies, i.e. ways to measure uncertainty/informativeness, in sequence labeling tasks is provided by Settles and Craven [54]. AL has been successfully applied to learning POS taggers [45, 61].

While AL has an overall positive effect in terms of reduced need for training examples (i.e. the annotator has to label less data), it has a negative effect on the performance of an annotator. The examples that are ranked higher by an AL

---

<sup>2</sup> <http://www.mturk.com>

<sup>3</sup> <http://www.wordrobe.org/>

selection scheme are in general more difficult to label and take longer to annotate [26].

In AL those examples are selected which are helpful for rapidly learning a classifier for a single annotation task. However it is not untypical to have multiple levels of annotation in a single corpus. This makes bootstrapping a corpus a multi-task annotation problem. Reichart et al. [44] extend AL from a single-task to a multi-task paradigm. In multi-task Active Learning the selection strategies of different learners are combined to identify examples which are most beneficial to all classifiers. In experiments with a two-task setting (NE tagging and parse tree annotation) they find that multi-task AL outperforms a random baseline and a one-sided standard AL baseline.

A refinement of AL that promises to further reduce the amount of manually labeled data is semi-supervised Active Learning [60]. The main difference to full AL is that in this approach instead of querying the oracle for a full sequence, e.g. a complete sentence, only a sub-sequence is presented for annotation. The underlying hypothesis is that a learner/classifier is particularly confident on large sub-sequences. If an annotator labels a full sequence (entire sentence), s/he would also provide labels for those sub-sequences that have little to no added training utility. One should note that in this work an annotator labels data rather than corrects labels to prevent a biased decision. Experiments on NE tagging showed when combining full with semi-supervised AL the amount of data labeled by a human can be greatly reduced while keeping a decent accuracy on a test set.

AL is mainly used to reduce annotation effort and, ultimately in a practical setting, annotation cost. However most research in this area focuses solely on reduction in training size (e.g. the number of items annotated) and assumes a uniform cost for all data. Ringger et al. [46] challenge this assumption and claim that annotation cost may differ for every datum. For example in POS tagging an annotator might need a varying amount of time per sentence depending on factors like sentence length and ambiguity. In order to measure true annotation cost they develop an ‘hourly cost model’ derived from data collected in an experiment on POS tagging. This model is used to predict the actual time needed to annotate a sequence of words. Such a cost estimator can be used to further refine the selection strategy of an AL algorithm. In a follow-up study Haertel et al. [27] seek to estimate the possible cost reduction by AL. By employing the previously developed ‘hourly cost model’ and simpler cost measures they compare different AL strategies in terms of their reduction in cost over a random baseline. In an experiment on POS tagging they find that in general high cost reduction can only be achieved when building a highly accurate model. Ultimately Haertel et al. [28] present a framework for including cost in AL algorithms. They claim that AL algorithms perform sub-optimally in terms of cost reduction if they select items solely based on their expected benefit and ignoring their true cost. By adding a cost heuristic into the selection strategies a more optimal trade-off between cost and benefit can be achieved.

Baldridge and Osborne [3] propound the hypothesis that AL has the inherent risk that the produced training data are hardly reusable with learners that are different from the one employed in the AL selection. In a study on learning parsers they found that created training data had low reusability value with other models and thus the advantages from AL are lost for these models. Tomanek et al. [61]

work on annotation of NEs contrasts this view. They suggest that reusability of the training data might be dependent on the problem setting and the type of AL method that was used.

### 3.3 Error detection in annotated corpora

Another type of research that is concerned with improving annotation is work that focuses on detecting and consequently correcting errors in annotation in existing resources. It stems from the problem that many corpora that are commonly used and widely accepted as accurate still contain a serious amount of errors. These errors in manually created or manually verified annotation persist, because even expert annotators make mistakes or annotation guidelines are too vague and allow for ambiguous interpretation. For example, the estimated error rate in POS annotation in the Penn Treebank is 3% [36]. This has implications for a number of different NLP tasks/studies. On the example of the Penn Treebank we see that the accuracy of a POS tagger tested on it can never reach 100% accuracy because of inconsistent annotation. Since the actions required for correction differ with the characteristics of the type of annotation used and it is typically the same as manual annotation, we will only present methods on the detection of errors. The work on this topic is not very diverse and mostly directed or exemplified on gold-standard POS annotated corpora.

A prominent approach is to find errors by inconsistency. It builds on the idea that if a token has different labels in identical contexts, this is an indication of an error. It works best for manually annotated or manually corrected corpora, since a fully automatically annotated corpus is rather consistent in the annotation and thus errors are more difficult to find. This approach can be implemented by searching for and listing n-grams of tagged tokens that differ in the label of the last token, so-called variation n-grams [21]. With this method Dickinson and Meurers [21] found 2495 variation nuclei of n-grams with length of up to 224 tokens in the POS annotation of the Wall Street Journal (WSJ) corpus. Another study that uses this and two other methods is given by Loftsson [35]. Next to variation n-grams they try to find inconsistencies in a POS tagged corpus of Icelandic by employing an ensemble of five different taggers and generating shallow parses. In their study the taggers are used to find instances where the ensemble agrees on a label but this label disagrees with the existing gold-standard annotation. An interesting outcome of the study is that each different technique of detecting errors gives a different set of error candidates. A large number of errors could only be found by one single approach respectively. Van Halteren [64] also uses a tagger to find inconsistencies in manually tagged text. Possible inconsistent instances are found by training a tagger on a corpus, tagging the very same corpus with the newly trained tagger and checking where the corpus and the tagger's output disagree.

## 4. Method

The central part of our work is to investigate how the gained knowledge of the Bits of Wisdom (BOWs) can be used and exploited to yield best effect. Since we learned correct labels from the BOWs and we want to get an improvement on the whole corpus, improving the models that are used for the automatic annotation is the obvious course of action. The way of improving the models consists in including the acquired information from the BOWs in the retraining process. Tagging the data in the corpus with an improved model will supposedly give an annotation with less or at least a different set of errors. The newly obtained data could then again be corrected. Figure 4.1 depicts how these steps can be looped to achieve a constant refinement of the data in the corpus. To date only the first step in this cyclic process, providing the corrected data, has been performed. With our work we want to add the next two steps in the loop: selecting new training data and retraining the part-of-speech (POS) tagger model.

Our main focus for our work on the selection step. We conduct experiments with two different general approaches to selecting data for retraining: *corrected self-training* and *Active Learning*. In both we make the same training data available and investigate how this data can be used best in this approach. Finally, the results of both are compared.

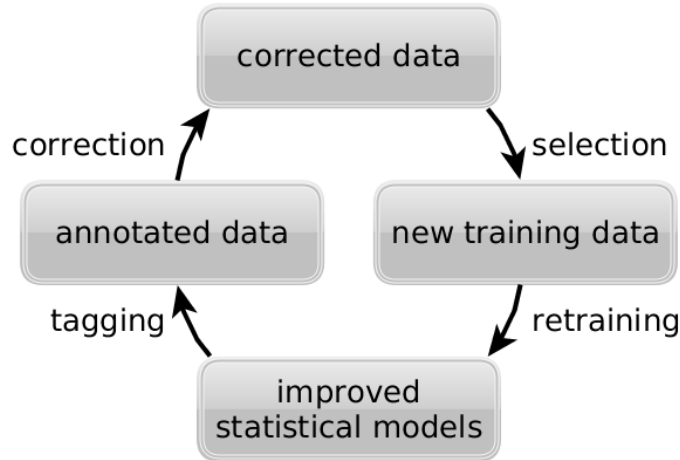


Figure 4.1: Schema of an iterative retraining process

The underlying automatic annotation in the GMB is based on the standard models of the C&C tools (see also section 2.1). For the POS level the standard model is trained on the training part of the CCGbank [29]. This model has a few limitations. For example, additional to the standard training, the authors of the C&C tools manually added support for quotation marks to the model by adding them to the dictionary of the model. Quotation marks are underrepresented in the training data and thus the standard model performs quite poorly when used for tagging them. Moreover since it is trained on newswire text exclusively it performs sub-optimally on non-newswire texts. The target of retraining is to create a model that makes less errors than the existing one.

BOWs are only applied for single tokens alone. However, as training and

tagging always needs a set of sentences rather than a set of words, it is the easiest to treat the sentence as smallest unit in our experiments. Therefore we can only expand our training data by full sentences with BOW-tokens. Adding a sentence might be problematic, as it possibly can include unwanted information (e.g. errors in annotation) and may negatively affect the results. This is a challenge to our experiments as we have to find a way to avoid those parasitic errors.

To insure comparability with the default model of the C&C tagger the parameters for training the models in the experiments have been set to the same values as the default model. This tagging model is a conditional ME model [17]. The model is defined by the following form:

$$p(y|x) = \frac{1}{Z(x)} \exp \left( \sum_{i=1}^n \lambda_i f_i(x, y) \right) \quad (4.1)$$

where  $x$  denotes a context,  $y$  a tag,  $f_i$  a feature,  $\lambda_i$  the corresponding feature weight and  $Z(x)$  a normalization constant. The ME framework requires that from all models which satisfy a set of constraints the most uniform model, i.e. the one with the highest entropy, is chosen. The constraints are that the expected value of each feature should match the observed expectation (estimated from training data). The constraints can be written in the following form:

$$E_p f_i = E_{\tilde{p}} f_i \quad (4.2)$$

$E_p f_i$  denotes the expected value of a feature  $f_i$  according to the model  $p$ :

$$E_p f_i = \sum_{x,y} p(x, y) f_i(x, y) \quad (4.3)$$

and  $E_{\tilde{p}} f_i$  the empirical expected value:

$$E_{\tilde{p}} f_i = \sum_{x,y} \tilde{p}(x, y) f_i(x, y) \quad (4.4)$$

The features of the model are binary-valued functions returning either 1 or 0 depending on a tag and a contextual predicate. For example, the following feature returns 1 if the current word is *that* and the tag is *DET*:

$$f_j(x, y) = \begin{cases} 1 & \text{if } word(x) = \text{"that"} \ \& \ y = \text{DET} \\ 0 & \text{otherwise} \end{cases} \quad (4.5)$$

The contextual predicate in this case is  $word(x)$ . The tagger is a reimplement of the MXPOST [42] and uses the same contextual predicates. These predicates include (among others) word and tag bigrams, single words and word prefixes [17].

For estimating the parameters we use the BFGS algorithm [39], as this was also used for the standard model. Similarly a Gaussian prior (0.707) was used for smoothing according to the standard model.

In this chapter we present the setup of our retraining experiments. We further introduce the data sets that were used for training and evaluation. In addition to evaluation on the POS level we describe how we assess the influence of retraining on higher levels of annotation on the example of the parser. Results of the experiments are presented in the following chapter.

## 4.1 Data Sets

### 4.1.1 Training Data Set

The training data consist of two parts: a fixed initial part and an additional variable part. The initial training data are taken from the WSJ part of the CCGbank, similar/according to the standard model of the C&C tools. The additional part is the data we get from the BOWs and thus taken from the GMB.

The initial training data includes sections 02 – 21 from the CCGbank. This training set comprises 39,604 sentences and roughly 930,000 tokens and is thus significantly larger than the set of sentences with BOWs. Since the tagging in this data set differed slightly from the tagset of the GMB, it had to be adapted. To adjust the training data to the target tagset, the tags *AS* and *SO* were replaced. Tokens that carried one of those labels were tagged with *RB* (adverb) instead. This design choice was motivated by the following reason: The CCGbank is built on the texts of the Penn Treebank and refined its POS annotation. The tags *AS* (applied to the word ‘*as*’ in the adverbial use) and *SO* (applied to the words ‘*so*’ and ‘*too*’ used as a submodifier) were newly introduced to the CCGbank and in all cases they only replaced *RB* tags. We are not aware of a documented motivation of this refined distinction of the *RB* tag. As this is a deterministic mapping, we are safe to revert those tags back to their original state.

To these data we want to add the information provided by the BOWs, i.e. data taken from the GMB. We leave out sections 00 – 09 from the GMB as held-out data and only use the remaining 90 sections. As we are interested in improving the existing models, the information will be added to the existing training data, rather than replacing it. This gives a richer representation of the available training data and also helps to keep the comparability to the current model. Since we chose the sentence as unit for our experiments, we consider all sentences that contain at least one BOW on the POS level for our experiments as possible training data. This results in a data set of 10,866 sentences with 262,101 tokens. These data contain 14,055 effective BOWs, i.e. corrected or verified POS tags. The held-out data is referred to in more depth in the next section.

### 4.1.2 Creating Silver Standard and Selecting Gold Standard Test Data

The performance of a tagger is usually measured by the accuracy of the correctly assigned tags on a given text [69]. This presupposes that a text with reliable annotation is given. Within the GMB, there does not exist any gold-standard part. As we want to do an evaluation within the GMB corpus and an evaluation on gold-standard data, we choose three different test sets: two external gold-standard and one within the GMB.

In order to test improvement within the GMB itself a test set was created, which we call ‘silver standard’. The goal of this test set is to measure the improved performance of a tagger in terms of corrected errors. Since the only labels we truly can be sure about are the ones provided by a BOW, we use those as our main source of measurement. We argue that tokens which had been incorrectly tagged by a tagger and required manual correction are in general more difficult and thus

more interesting cases. Thus evaluating changes on them can give an insightful picture about the data in general. Moreover, this test set gives the possibility to get a more specific measurement how retraining impacts the annotation in the GMB. Since we measure on identified errors, we get a direct estimate how many errors within our data get fixed by improving the tagging.

The silver standard was created out of the first 10 sections of the GMB and contains 6,483 sentences and 142,344 tokens. Within the GMB silver-standard test set there are 2,917 effective BOWs, i.e. corrected or verified POS tags, which equates to 2.04% of the tokens.

We chose those sections as a test set since they had a slightly higher rate of BOWs per token than the rest of the corpus. We further increased the value of this test set by verifying the existing annotation and adding more BOWs. Building on the assumption that the overall quality of the POS annotation is reasonably correct, we did not verify each sentence individually, but rather tried to find errors in annotation automatically. Possible errors in tagging were found by disagreement of two models. By training a supposedly improved model and retagging the sentences in the test set, it was possible to find instances where the new and the old model disagree. In our case the old model is simply the default model that provided the initial annotation. The newly trained model includes all sentences that contain at least one BOW of the remaining 90 parts of the GMB in addition to the original initial training data. By this means, more than 1,000 disagreements were identified.

These disagreements were manually checked and the correct tags assigned. An advantage of this method is that not only brings up erroneous annotation but existing annotation got also verified. In the process of the manual assessment, the annotator was presented one sentence (without annotation) at a time and the target word (the word with unclear labeling) was highlighted. The annotator could choose from the two labels proposed by the taggers or overwrite with a different tag. If necessary s/he could also access the context of the sentence, i.e. the rest of the document.

Finally, it is noteworthy that our definition of silver-standard corpus differs from the definition given by the creators of the CALBC corpus [43]. The main difference is that the silver standard of the CALBC project is created by employing a sophisticated harmonization method on the output of multiple automatic systems to get the final annotation rather than employing human annotators.

The biggest drawback of the silver standard described above is that it does not allow any statement about the quality of all those changes in tagging that are not covered by BOWs. This means that it is possible to measure how many errors get corrected, but it does not give reliable information whether new errors are introduced. To address this problem we need gold-standard data, i.e. data that have fully verified annotation.

As the CCGbank corpus provided the initial training data, it is logical to also use its test section as a gold-standard corpus for testing. We use sections 22 – 24 of the WSJ part (consisting of 126,751 tokens) of the CCGbank as test data. The text in the WSJ corpus is purely newswire text. Considering the fact that state-of-the-art POS taggers, including the tagger used in this work, have reached an upper bound in accuracy (that hardly is surpassed by new methods) in this particular genre, no significant increases in accuracy are expected on this test set.

Similar to the initial training data, we also replaced all *AS* and *SO* tags with *RB* in this test set to match the target tagset.

This outlook and the fact that adding sentences from the GMB to the training corpus also means adding different genres encourages the use of a gold standard with more diverse genres. Changes in performance of the tagger might be more easy to quantify on non-newswire text. Furthermore, since the GMB incorporates multiple genres it is of particular interest to measure performance on a variety of genres. The Manually Annotated Sub-Corpus (MASC) [30] of the ANC project<sup>1</sup> has been identified as a suitable resource. MASC includes 19 different genres and more than 500,000 tokens of written text and transcribed speech. Annotation on the POS level in this corpus was automatically created and manually validated. We use the full corpus in our evaluation.

Even though the MASC has a part consisting of newswire text (including some WSJ documents) we are convinced that the use of the WSJ test set in addition to the MASC is advisable. This helps to ensure that the especially high performance on this particular genre is not derogated by possible negative influence of the added training data.

This gives three test sets for evaluation and we summarize their key characteristics in Table 4.1:

Table 4.1: An overview of the three different test sets used

Name	Type	Size in tokens
MASC	gold standard	500k
WSJ	gold standard	126k
GMB	silver standard	142k, thereof 2,917 BOWs (2.04 %)

## 4.2 Sampling strategies for selecting additional training data

By retraining the statistical tagger with a model that includes data from the corrected corpus, we strive to make the annotation on the whole corpus better. Our working hypothesis is that, when correcting an incorrectly tagged sentence and adding it to the training data, the model will take up this correction and label similar instances correctly in the future. Since the tagger is (partly) trained on its own output, this retraining process is related to self-training [16]. In contrast to pure self-training, where no human agent is involved and a tagger is trained iteratively on its own predictions, in our case the output of the tagger is corrected before applying it in training. As there is no guarantee that sentences are corrected in their entirety and thus there is still some semi-supervised element to it, we refer to our technique as *corrected self-training*.

By exploring different ways of adding the gained knowledge of the added BOWs, we want to address the question of how to retrain the statistical model effectively. We want to be able to choose a subset of the corrected sentences for retraining purposes that provides the optimal improvement for the tagger. By

<sup>1</sup> <http://www.anc.org/>

increasingly adding more sentences to the training data and iteratively training and evaluating the tagger, we can measure the change of its performance with varying amounts and subsets of additional training data. We will investigate three different selection strategies. The success of the different selection methods also gives information on how to minimize supervision of the retraining process. For our experiments we iteratively retrain and evaluate after adding 200 sentences to the training data. This allows a decent trade-off between reasonably small steps and not too many training iterations.

### 4.2.1 Random Sampling

For our baseline no special selection method is used. Examples are taken at random (without replacement) out of the pool of available sentences. In order to validate the results of the baseline, results are averaged over five different samplings.

Random selection is a common baseline and gives the possibility to compare results of more sophisticated selection methods to pure chance. It represents retraining without any supervision.

### 4.2.2 Longest sentence first

Another baseline that requires only minimal supervision builds on the simple supposition that longer sequences contain more information, since they contain more (labeled) tokens. As more information also means a potentially higher training utility, sentences get selected by their length starting with the longest. We measure sentence length in number of tokens per sentence. This selection method was for example used as a baseline in studies presented by Ringger et al. [45] and Haertel et al. [27].

### 4.2.3 Cautious sampling

The biggest threat to an effective retraining process is adding flawed data. As described above, there is no guarantee that a sentence, which contains at least one corrected tag, is correct in its entirety. An example that contains incorrect tagging could be harmful in retraining. To ensure reliability in retraining, only faultless examples should be considered while faulty examples should be disregarded. However, there is no definite means to tell whether a sentence has completely correct annotation. A simple way of estimating correctness of a sentence is by its ratio of corrected tags. The ratio is easily calculated by dividing the number of corrected tags by the number of all tags in a sequence. Building on the assumption that all provided BOWs are correct, this measure can also be seen as an indication of the probability that a sentence contains an incorrect tag. A higher ratio implies a lower probability. We argue that a sentence with a high ratio of BOWs has received high amount of attention by at least one annotator. Since humans consider the context of a token when deciding the correct annotation, it is likely that other flaws in the annotation are spotted and corrected. This hypothesis augments the BOW ratio measure and renders examples with a high ratio unlikely to be incorrectly tagged.

With this selection method examples with a higher BOW ratio are added first, since those are less likely to contain errors. In cases of equal ratio longer sentences are preferred. In our data the mean ratio of BOWs per sentence is 0.059 with the median at 0.0476. In the sentence with the highest ratio half of the tokens have corrected tagging. In the sentence with the lowest ratio one out of 72 labels is provided by a BOW.

### 4.3 The Active Learning approach

Active Learning promises to speed up the training of a classifier, or in our case a tagger, by finding the items with the highest expected training utility in a set of (unlabeled) data. As it also consequently reduces the amount of labeled training data, that is needed to build a robust classifier, drastically, it is most commonly used in bootstrapping data sets where only a small initial training set is available. Both of these two characteristics are desired when (re-)training tools for annotation.

With simulating AL we want to extend our approach to investigating the effect of retraining a bit further. AL adds the notion of redundancy to the retraining process as it prefers items with a high informative value. This means that it is possible to find a subset of BOW-sentences which already has the same or at least comparable informativeness and retraining value as the full set. Outcomes of the experiments indicate whether AL would have helped to reduce the manual effort invested in providing corrections by decreasing the amount needed. Another point we might find is that since we already have an initial training set of significant size and AL is most effective in early stages of bootstrapping [45], we might also find that bootstrapping approaches have insufficient effect in the retraining. Ultimately we can draw conclusions whether AL is a suitable way of directing future correction efforts.

In the AL framework data instances in a pool of unlabeled examples are rated by their expected training utility. For the ones with the highest estimate an oracle can be queried for the correct classification (or labeling) of the instance. There are a number of different measuring techniques to assess the training value, i.e. informativeness, of a datum. A general way to estimate expected training utility of a sequence is by measuring the uncertainty of a classifier. Two very common algorithms to get the uncertainty are Query by Uncertainty (QBU) and Query by Committee (QBC) [27, page 2]. Both have successfully been applied to POS tagging [45]. We use both in our experiments and their theoretical aspects are described in the following subsections. In a real world setting the oracle, that provides the true labels for a queried sentence, is typically a single human annotator, that is assumed to be flawless, but it can also be multiple annotators that are not unerring [28, page 2]. Since employing an actual human agent is not feasible for our study, we use the BOW database as the oracle. We argue that this is very close to an actual oracle of human annotators. Our working assumption is that a sentence that carries at least one BOW has a high probability of being correct.

We run the AL experiments in two different configurations of the pool of instances that is queried by the AL selection method. First we use the same set of sentences that we used in the retraining trials, i.e. all sentences containing at

least one BOW. Secondly we limit the pool in this data set to 50% of sentences that have the highest BOW ratio. By this we want to exclude examples that might still contain errors. This helps to introduce the same notion that we followed with sampling by BOW ratio (see section 4.2.3) into the AL experiments. Namely, we want to prefer correct examples that have a higher expected training value. Consequently, since less data are available in the latter configuration, fewer iterations are run. In the experiments we use all available data exhaustively. That means that no stopping criterion is defined and we use all sentences that are available to us in the pool.

Commonly stopping criteria are designed to take effect at the point where obtaining more labeled data (selected by the employed technique) loses its effectiveness and thus results in higher resource spending. This can also be the case when the maximum accuracy possible for the classifier is achieved, e.g. a measured plateau on a test set. Additionally a number of criteria have been proposed, that are based on the concept of an intrinsic measure that signals decreasing usefulness by exceeding a certain threshold. In practice, however, AL is often stopped by external factors, like resource limitations [53, p. 77].

As we have a rather small pool available we chose not to use any stopping criterion. Since we only simulate AL, we don't have any limitations other than data limitations. Additionally, this allows us to see whether strong deterioration effects occur by not stopping.

An important parameter to the AL algorithm is the number of elements added to the training data in each iteration. In an ideal setting sequential selection (one item is selected and added to the training data in each iteration) is executed. In practice, however, batch selection (adding a number of items per iteration) is more commonly used, as it allows for much faster running time of experiments by significantly reducing the number of iterations. A number of studies employing AL [45, 2] found no significant difference in performance when varying batch sizes. For our experiments we use batch selection with a batch size of 200 sentences. Similarly to the corrected self-training experiments, this allows a decent trade-off between reasonably small steps and not too many training iterations.

### 4.3.1 Query by Uncertainty

Query by Uncertainty (QBU) is a measurement of informativeness that stems from the general framework of uncertainty sampling [34]. The quintessence is the idea that the training utility of an instance can be estimated by the classifiers uncertainty. In the case of POS tagging, the output of a single probabilistic tagger containing all possible tags with their associated probabilities is used. Sentences on which the tagger is unsure have possibly a high training utility. Entropy [56] is a useful measure for this uncertainty as it gives the information contained in the probability distribution according to the tagger's model. Specifically we use the token entropy (TE) [54] of a sentence  $x$  which is calculated by the following formula:

$$\text{tokenentropy}(x) = -\frac{1}{T} \sum_{t=1}^T \sum_{m=1}^M P_{\theta}(y_t = m) \log P_{\theta}(y_t = m) \quad (4.6)$$

For every token  $t$  in a sequence with length  $T$  it sums the entropy of the probability distribution of possible labels in  $M$  for  $t$ . The marginal probability that  $m$  is the label for  $t$  according to the model  $\theta$  is denoted by  $P_\theta(y_t = m)$ . By normalizing by the sentence length it is avoided that simply longer sentences, that would otherwise have higher entropy values, are preferred.

### 4.3.2 Query by Committee

The Query by Committee (QBC) framework was proposed by Seung et al. [55]. Freund et al. [24] provide a in depth analysis of it and Argamon-Engelson and Dagan [2] apply QBC together with HMMs in POS tagging. It is sometimes also referred to as a Monte-Carlo technique.

In QBC the possible labels and their associated probabilities are provided by a committee rather than a single classifier. The committee consists of an ensemble of classifiers that all vote on a classification. The disagreement of the classifiers (or in our case: taggers) can be used to identify difficult cases. Members of the committee can be selected in different ways. For example, the members can be chosen at random from a set of classifiers. Another possibility would be to combine different algorithms to get diversity among the members. There are different approaches to how to split the training data among committee members. It can be sampled with or without replacement. Outcomes of QBC are also influenced by the size of the committee. Small committees have proven to produce results comparative to those produced by bigger committees [45, 2].

In order to get a diverse ensemble of taggers we train the taggers with different subsets of the available training data. For our experiments we sample the training data with replacement and make 80% available to each member. By this we aim to ensure that the taggers maintain a fairly high accuracy, which is needed to avoid disagreement caused by insufficient training. It also increases the chance that each committee member gets a share of the newly added examples. We use a committee with three members, i.e. three different models.

To measure disagreement among the members of a committee  $C$  on a certain sequence  $x$  we use vote entropy (VE) [54], which is defined by the following formula:

$$voteentropy(x) = -\frac{1}{T} \sum_{t=1}^T \sum_{m=1}^M \frac{V(y_t, m)}{C} \log \frac{V(y_t, m)}{C} \quad (4.7)$$

where  $V(y_t, m)$  denotes the number of committee members that voted on label  $m$  for token  $t$ .

## 4.4 Impact on higher levels of annotation — (Syntactic level)

Improving the annotation on one level in the corpus has an influence on all dependent levels. Annotation is interdependent since the input features for a number of classifiers are the output of another. For example, the parser uses the output of the POS tagger and the supertagger. As a consequence incorrect annotation can result in cascading errors in NLP pipelines [23]. But also the opposite is true:

“even modest improvements in POS tagging accuracy can have a large impact on the performance of downstream components in a language processing pipeline” [19]. We are interested how our retraining efforts influence the results of other parts in the pipeline. One of the integral parts of the pipeline is the parser. Following the proposition that improved performance on the POS level positively influences the performance of subsequent processing, the parser should benefit from an improved POS model.

Since no gold-standard parsed data exists within the GMB we can only estimate the improvement. One performance measure of a parser on unannotated data is its coverage [11]. The coverage is defined as the percentage of sentences that were successfully analyzed by the parser. Ideally the coverage of the parser would increase with improved POS tagging. A decreased coverage could be an indication of newly introduced errors and decreased consistency in the tagging. Even though coverage is a weak indicator it will help to show whether there is a performance change on our main data set. Similar to the other experiments this evaluation is done on our GMB test set.

Additionally we evaluate the influence of an improved POS annotation on the parser on a gold-standard test set. For this we use the WSJ test set from the CCGbank. The produced dependency output from the parser is compared to the gold-standard data. The CCG parser produces the dependencies as a 4-tuple: a head of a functor, a functor category, an argument and a head of an argument [15]. We calculate precision ( $p$ ) and recall ( $r$ ) for these dependencies while differentiating between labelled and unlabelled dependencies. We further calculate the F-measure as follows:  $(2 * p * r) / (p + r)$ . For a labeled dependency to be matched correctly, the entire 4-tuple has to be matched. An unlabeled dependency is matched if the heads of the functor and the argument appear together in some relation [15].

## 5. Results and Discussion

In this chapter we present and discuss the evaluation results of our retraining experiments that were outlined in the previous chapter. We evaluate different sampling strategies and compare those to two AL strategies. In both retraining settings the default model is enhanced by data taken from the BOWs. Finally, we assess the influence of a retrained POS tagger on higher levels of annotation.

For completeness we first present the results of the current baseline. The default model trained on the original training data serves as the point of origin for our experiments. The results produced by this model are given in Table 5.1. Not surprisingly it achieves a very high performance on the WSJ test set and a decent performance on the MASC. As one can expect in a contrast to that, the rate of predicted labels that match the true label chosen by the annotator (matched BOWs) in the GMB test set is rather low. Recall that in the GMB test set we try to measure performance in terms of corrected errors or rather matched manually verified labels. Thus the number of errors the baseline model makes is high since all corrected errors are the ones corrected by this model. The number of BOWs already matched BOWs by the baseline, however, can be attributed to the fact that not all BOWs in the test set are corrections. Some of the BOWs provided in the creation of the GMB test set verified existing annotation.

*Table 5.1: Results of the default model on the three test sets*

	Baseline
Accuracy, WSJ	96.914%
Accuracy, MASC	90.086%
Percentage of matched BOWs, GMB	15.05%

### 5.1 Random, Longest Sentence First, & Cautious Sampling

The results from retraining after iteratively adding sentences with BOW-tokens to the training data are given in Figure 5.1. We compare three different selection strategies: (i) random selection, (ii) adding sentences by their length in tokens starting with the longest (Longest Sentence First) and (iii) choosing sentences with a higher ratio of BOWs before others (BOW Ratio). The graph does not include performance on both gold-standard test sets (WSJ & MASC) since the accuracy for those only changes insignificantly and no notable improvement or decline was evident over the course of retraining. Testing on different subsets of MASC with varying genre combinations to see, if at least partial improvement has been achieved, showed similar results. A change in performance, however, is clearly measurable on the GMB test set that gives an estimate of improvement in terms of matched BOWs, i.e. the true labels that were predicted the same by the tagger.

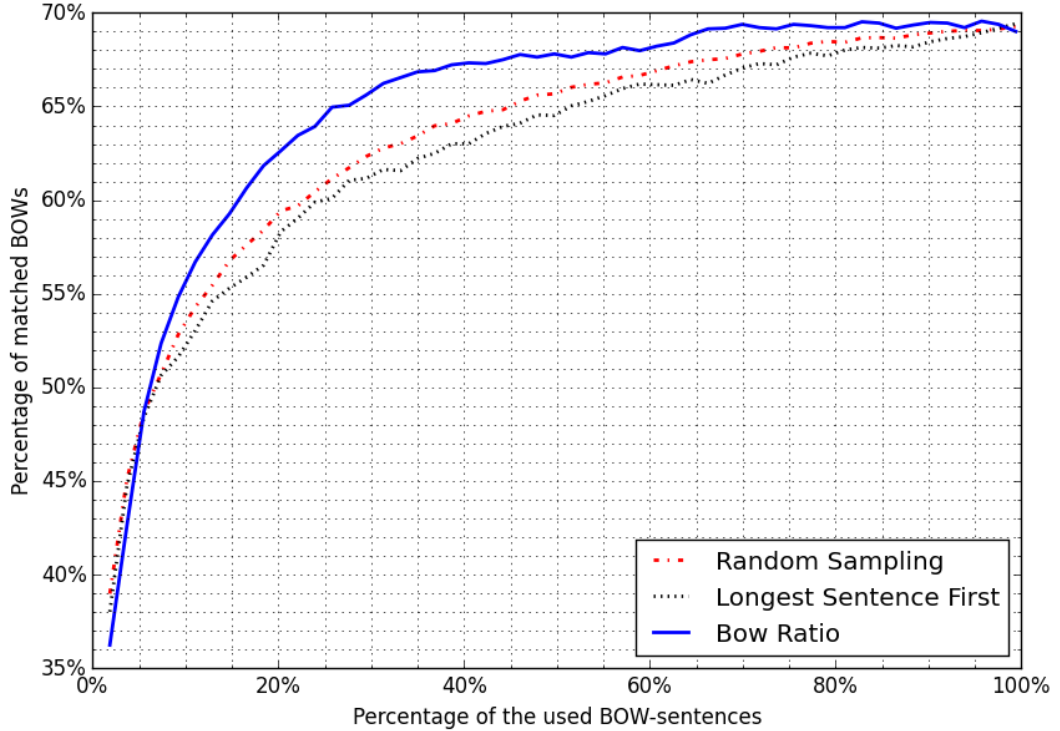


Figure 5.1: Performance change with increased training size by three different selection methods measured by matched BOWs

We see from the graph that all three methods achieve good results on the GMB test set and for all of them the biggest gains are made in the early re-training iterations with only a small subset of the data used. Already the first 200 sentences (first iteration) give a significant improvement over the baseline model. With this first iteration already more than twice the amount of BOWs in comparison to the baseline are matched. The rapid improvement rate already slows down after only a quarter of the data is used. A ceiling is hit at roughly 70% of matched BOWs. A possible explanation for this is the notion that the utility of the training data is exhausted, i.e. the information needed to correct the persisting errors is not present in the training data. The simplest selection method, random sampling, already shows decent results. Longest Sentence First performs worse than the random selection. Selecting by BOW ratio outperforms the other two.

BOW ratio perhaps not only describes how unlikely it is that a sentence still contains errors, but can also be seen as a measuring tool of training utility since sentences with a higher rate of mislabeled cases add more information about correction in retraining.

The best model (chosen from the results on the GMB test set) achieves 69.56% of matched BOWs with 96% of the available sentences added. A comparison to the baseline is given in Table 5.2. This best model uses almost all data available and performs marginally better than other models using less data. As described above performance improvement stalls with increasing amount of the added training data. A model for example that only uses 70% of the BOW-sentences performs on a par with this model and matches 69.39% BOWs.

Table 5.2: Comparison of the results of the default model and the best model chosen from the corrected self-training on the three test sets

	Baseline	BOW Ratio
Accuracy, WSJ	96.91%	96.95%
Accuracy, MASC	90.09%	90.24%
Percentage of matched BOWs, GMB	15.05%	69.56%

Since the GMB test set is our strongest measure of improvement and it is no gold standard, it is of special interest to see how big the impact of retraining is on the annotation that is not verified by a BOW. Figure 5.2 shows how the amount of changes in labeling not covered by BOWs varies with the training size. We see that only a very small percentage of the labels that are not verified differ. This reassures us that the results measured with the GMB test set are meaningful.

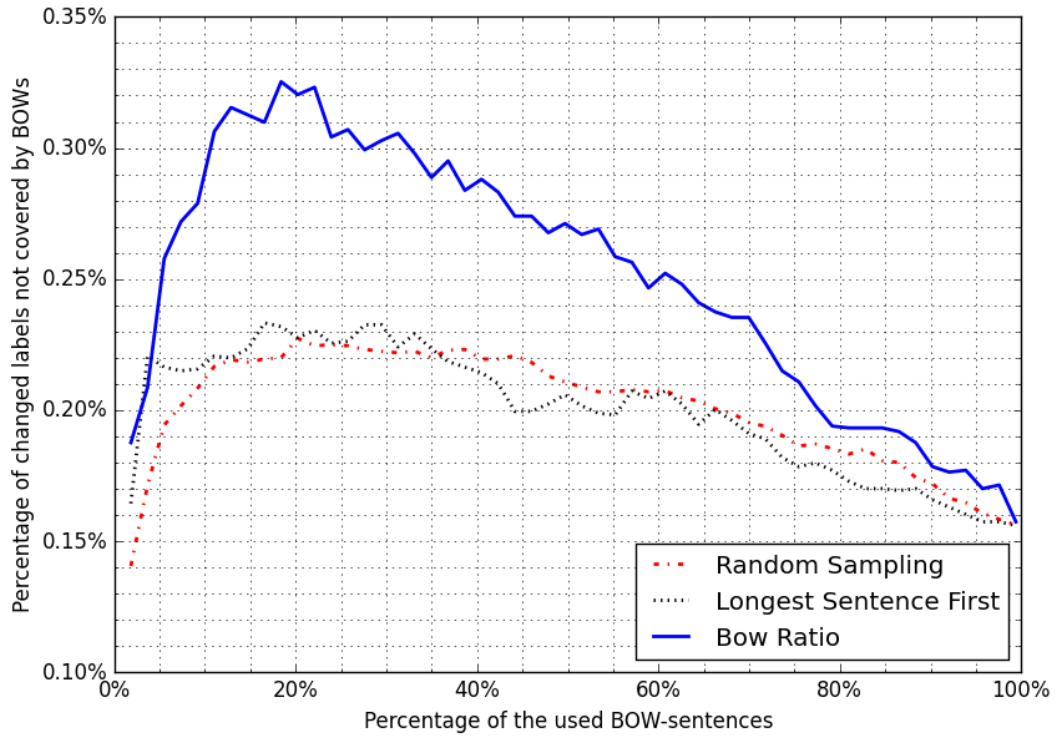


Figure 5.2: Percentage of changed labels that are not covered by a BOW with increased training size

The number of changes increases in the early stages of retraining and decreases with a growing amount of added training data. Retraining by BOW ratio triggers the highest rate of changes. The low number of changes in the final stages describes the fact that the new annotation is close to the original. This could have two reasons: On the one hand, assuming that the original annotation is correct it would be a sign of stabilized retraining. As more data get added, the newly learned information gets refined by more examples and thus the model makes less errors. This view is supported by the fact that no performance drop was measurable on the silver nor on the gold standard test sets. On the other hand, if inconsistencies in the training data are assumed (i.e. not all or too few

instances of a characteristic error are corrected), it could mean that misclassifications are learned. In this case the corrections lose their effect in retraining. The fact that the score of the BOW-measure was capped in the endmost rounds could be evidence of inconsistencies in the training data and hence endorse the latter interpretation.

If one were not to choose the best, but rather a model with the greatest impact on annotation, the rate of improvement would need to be weighed up against the amount of changes. In other words, the model performance could also be optimized for producing the most changes in output with the least amount of data added to the input possible. Such a model might be desired to find a large quantity of changes that can be checked by annotators, while maintaining an acceptable improvement rate. It could provide a way to get good results before deterioration effects start to show. For example in our case, a model that uses 25% of the additional training data can give a decent improvement over the baseline (almost 65% of matched BOWs) and provide a bigger amount of so far unchecked labels than the further refined models.

## 5.2 The Active Learning approach

Similarly to the results from the corrected self-training experiments presented in the previous chapter, results on the gold-standard test sets show no significant changes for all experimental conditions with the AL approach. We compare two different conditions of AL: (i) making the full set of BOW-sentences available and (ii) limiting the set to the upper half of all BOW-sentences sorted by BOW ratio. In both conditions we compare the two selection methods (QBU & QBC).

We see from Figure 5.3 that AL with all available BOW-sentences is not better than selecting by BOW ratio, but the performance is above random selection with the exception of the very first rounds. The selection method QBU performs significantly better than QBC in the first few iterations (with 10% – 30% of the additional training data used). Not surprisingly AL also only reaches scores up to 70% on the GMB test set.

When using the limited set of additional training data, the performance of the AL selection exceeds the performance of AL with the full set. The graph given in Figure 5.4 shows that both QBC and QBU achieve slightly higher rates in matched BOWs with less training data. This supports the assumption that the training data, that were excluded from the selection (the sentences with a low ratio of BOWs) in the latter configuration of AL, do contain errors. This would explain the fact that the sentences that were deemed valuable and got selected in the first few iterations with the full set, do not achieve the same performance as the sentences that got selected from the reduced set. Again QBC is in general inferior to QBU, but the difference is smaller in comparison to the first configuration, as QBC manages to outperform QBU in the very first iterations. The second configuration of our AL experiments does not only give a better performance but also a higher rate of changes in labels not covered by a BOW in GMB test set. From Figure 5.5, that gives the changes in the second configuration, we see that the two AL methods trigger less changes than selection by BOW ratio. This is remarkable since they achieve a score comparable to BOW ratio selection.

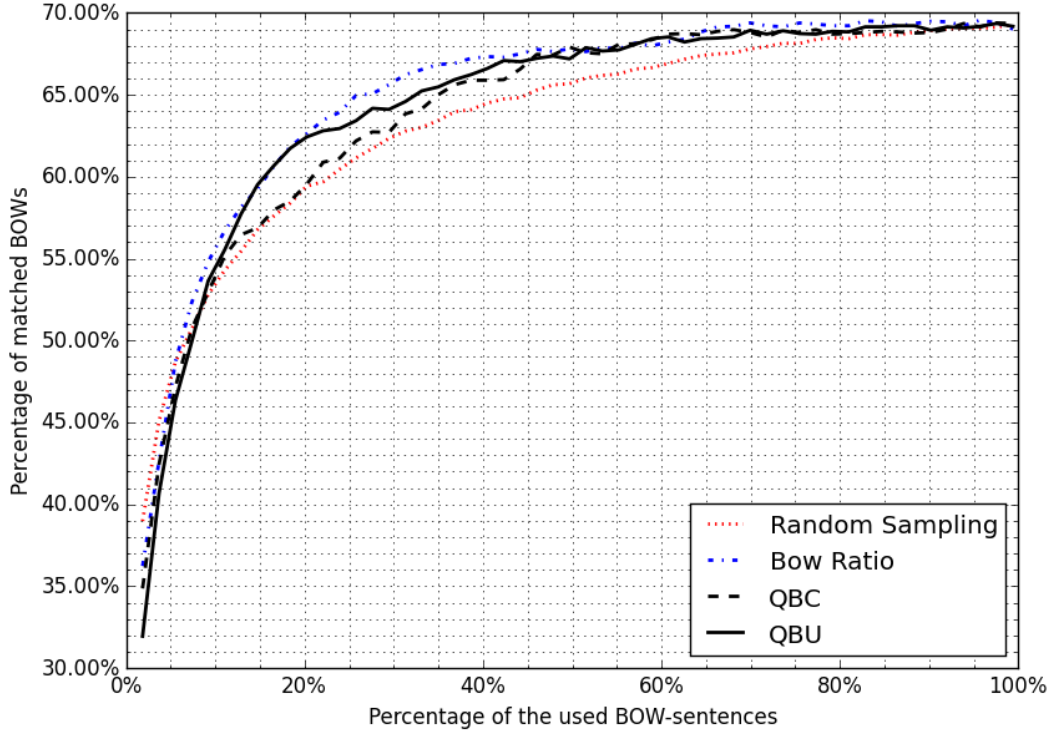


Figure 5.3: Performance change with increased training size by QBU and QBC measured by matched BOWs (all sentences with BOWs available)

The effect of AL becomes especially visible when comparing the amount of data needed to reach a certain level of performance. When looking at the maximum performance achieved with retraining (Figure 5.1), we can assume 70% of matched BOWs as our upper bound. To surpass 90% of that upper bound (63% of matched BOWs), i.e. to get a reasonably good performance, the different selection strategies require different amounts of added training data. Figure 5.6 gives a comparison for four different selection methods. QBU (using the second configuration) requires the least added sentences. Random selection and Longest Sentence First need significantly more with almost up to twice the amount. Selecting by BOW Ratio only requires slightly more than QBU. This shows how selective sampling can help to reduce the amount of data needed for effective retraining. As a conclusion this suggests that AL can indeed be of help to reduce the correction effort needed by excluding redundant elements from correction.

When interpreting the results of the AL experiments we have to keep in mind that results are not expected to be the same in a real world setting. By simulating the human oracle and limiting the pool to a set of sentences that already have a correction, we make our experiments artificial and the outcomes are influenced by this. There is the possibility that performing AL in a real world scenario gives better results and/or shows stronger effects. For example, with the whole corpus available for selection it might show that so far uncorrected sentences have a higher training utility. Effects of AL on error correction might also be boosted, because a human annotator is required to correct the full sentence rather than single tokens only and thus the training data are considered to be almost error-free.

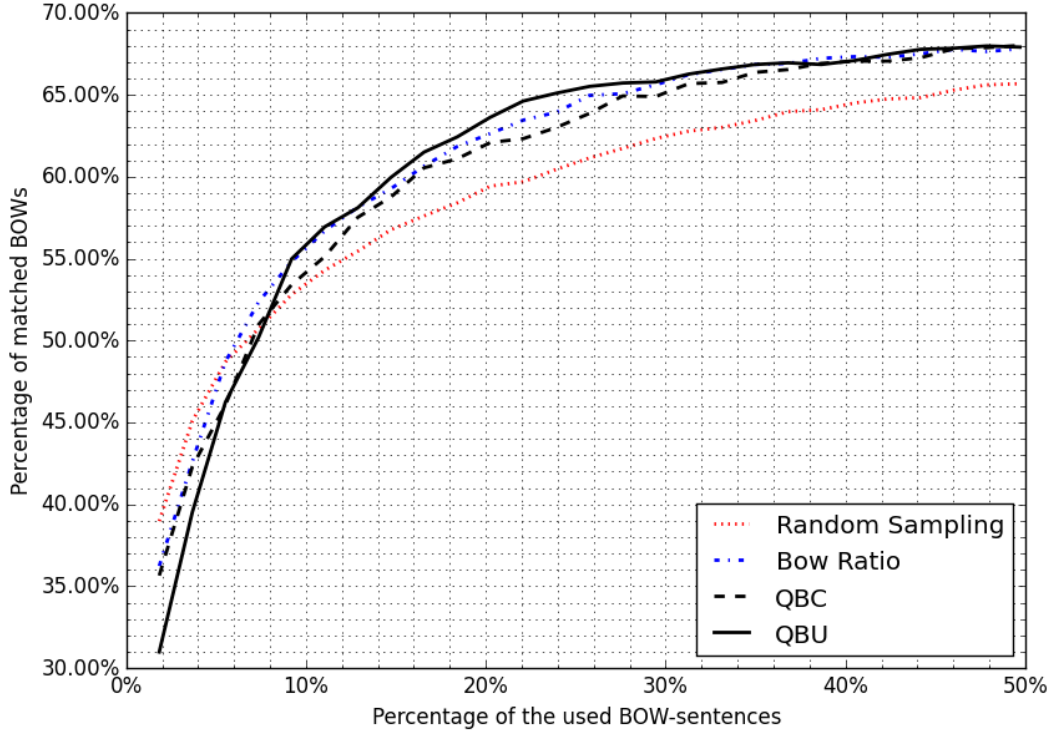


Figure 5.4: Performance change with increased training size by QBU and QBC measured by matched BOWs (reduced set of sentences with BOWs available)

As the sampling of the training data that are selected for training the committee members in QBC is random, results might differ when testing again. Since QBU and QBC, however, show similar results we are confident that the difference for a varying committee will not be significant.

A risk of AL is that the selected training data are specific to the type of classifier used and have low re-usability with other classifiers. Since the training data which are produced by AL are only a small part of the overall training data, we consider re-usability ensured. However we identified another risk in our setting of AL. A characteristic of our added training data is that they were corrected on single tokens only. If sentences that are selected for training still contain errors in annotation, it might be problematic for future AL iterations since cases with similar errors might not be selected as the uncertainty on those is already reduced. This means that relying only on AL might make an annotator blind to certain errors if the training data is erroneous.

While the two selection methods presented here (QBC & QBU) are the most common within similar problem settings and are thus used to exemplify AL, it is noteworthy that there do exist other approaches, for example query-by-model-improvement [47, 1]. Moreover, within the two frameworks a number of different ways of uncertainty measurement have been used to tune the properties of the method.

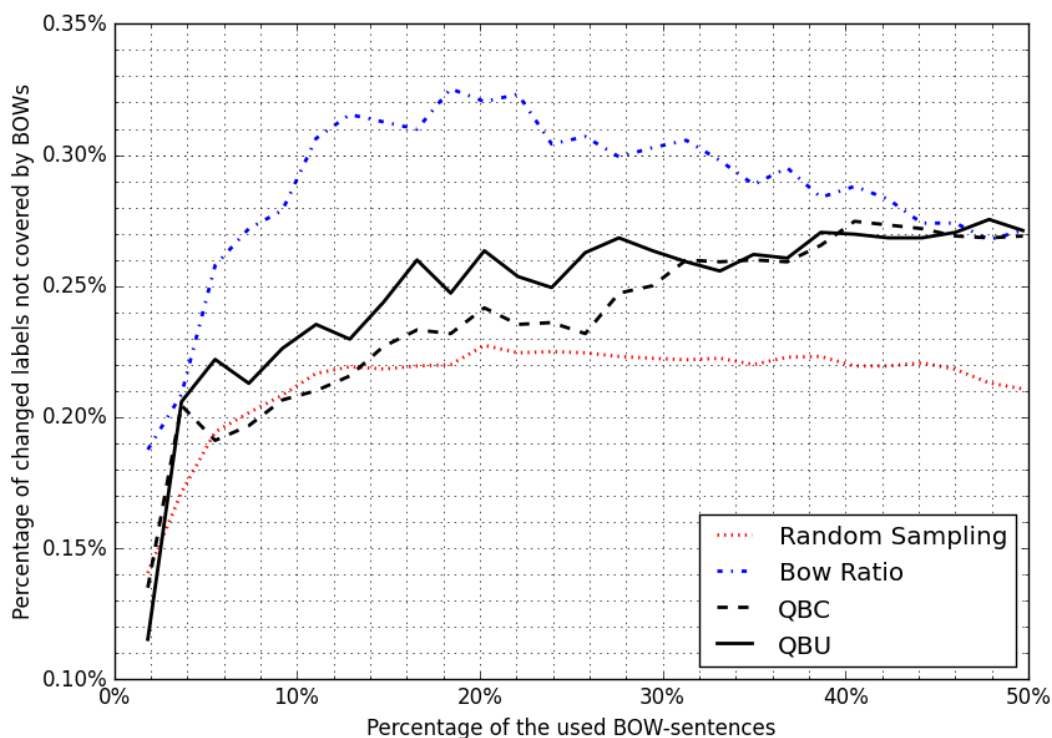


Figure 5.5: Percentage of changed labels that are not covered by a BOW with increased training size (reduced set made available to the AL selection methods)

### 5.3 Impact on higher levels of annotation — (Syntactic level)

We evaluate the influence of POS annotation on the parser on the GMB test set and on the WSJ test set. Our main source of measurement on the GMB set is the coverage of the parser. Due to the unvarying performance of the POS tagger on the two gold-standard test sets, no significant change in parser coverage arose on those test sets. We give dependency precision and recall scores for the WSJ test set.

When testing the parser on different POS tagged versions of the GMB held-out data, changes in coverage were observable. In Table 5.3 we compare how big is the influence of changes in the POS annotation in the input data to the parser in terms of coverage on the GMB data. With the annotation of the baseline model, very high (almost full) coverage is already achieved. When employing the model that achieved best performance in retraining, the coverage can be raised. The improvement is only marginal (31 more sentences could be parsed). The highest coverage is achieved when using the current annotation in the test set with all available BOWs. This suggests that the best model after retraining, while it reproduces almost 70% of the BOWs in the data (see Table 5.2), does not fix a number of major errors in annotation that cause the parser to fail. Another possible explanation would be that the newly trained model introduces new errors which obstruct the parser. Those errors could be identified and corrected in subsequent retraining iterations.

Evaluating the influence of improved POS annotation on gold-standard data

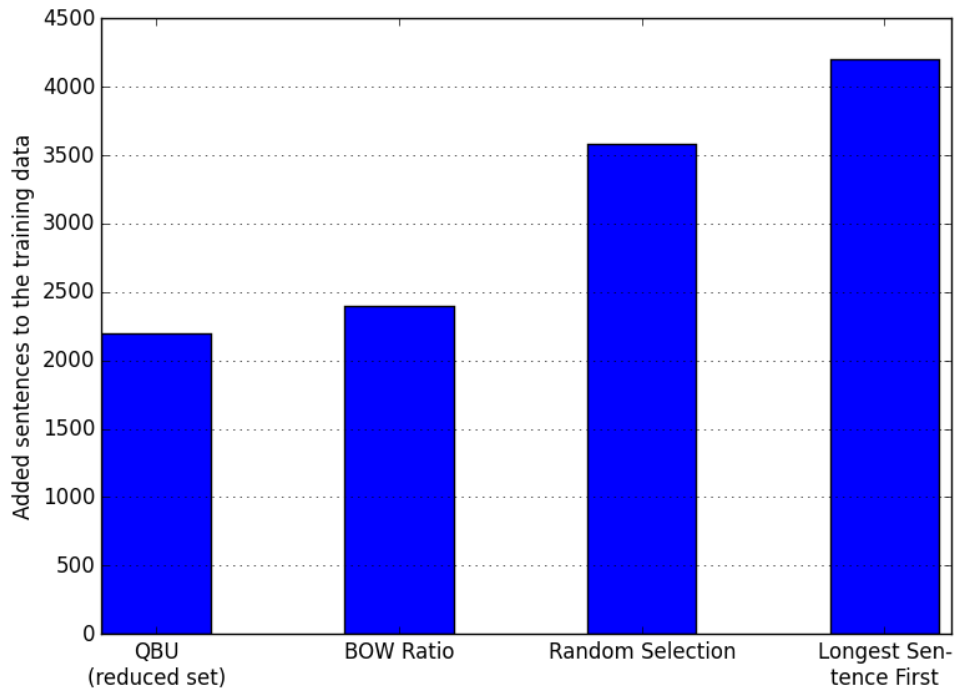


Figure 5.6: Number of sentences added to the training data to achieve 63% matched BOWs on the GMB test set

Table 5.3: Comparison of influence of different sources of POS annotation on the parser coverage

	Baseline model	Best model after retraining	All provided BOWs
Coverage on the GMB test set	99.34%	99.46%	99.68%

gives a similar picture to what we saw on the GMB test set. As we can see from Table 5.4 the performance of the parser improves with better POS annotation. The baseline model has the lowest scores. Our hypothesis that an improved POS tagger (represented by the best model we chose) has a positive influence on the parser is supported by the fact that the retrained POS tagger model achieves scores that are higher or at least on par with the baseline. However only a slight improvement is visible. The (manually verified) gold-standard annotation gives the best results for the parser. Not surprisingly, the scores for unlabelled dependencies are higher in all three instances.

Since we saw no significant improvement of a retrained tagger on the POS level of the gold-standard data itself, it is not surprising that no significant improvement is visible on the syntactic level. Another possibility of assessing the influence of adapted POS annotation on the syntactic level, would be to evaluate changes on the CCG supertags.

Table 5.4: Evaluation of the parser on gold-standard data with different sources of POS annotation

		Baseline model	Best model after retraining	Gold-standard POSS
labeled depen- dency	Precision	84.99%	85.01%	86.85%
	Recall	82.97%	83.07%	85.19%
	F-measure	83.97%	84.03%	86.01%
unlabeled depen- dency	Precision	91.64%	91.67%	92.80%
	Recall	89.46%	89.57%	91.03%
	F-measure	90.54%	90.61%	91.91%

## 5.4 General Discussion

It is not surprising that the performance on the WSJ test set could not be increased with retraining as it is already at the state-of-the-art level. Moreover, since also no decrease in performance was visible it is an indication that the added training data are not harmful. POS tagging is known to perform best when training and test data are very similar [25]. Following this proposition we hoped to increase the performance on newswire text genres tested with the MASC corpus. However, also no significant improvement was measurable. An explanation for this could be that the difference between the corpora is still too big for our retraining to show effect. For example the MASC contains a number of genres (e.g. transcribed speech, e-mails, Twitter, ...) that are not included in the GMB.

When selecting sentences according to their expected correctness of annotation we only consider the ratio of BOWs in the sentence. However, there might be other indicators within our data, for example the number of different annotators that provided corrections for a sentence. Additionally, we could assume a sentence with many BOWs on higher levels of annotation is likely to be correct on the POS level since the annotator would probably have corrected this level as well otherwise. We also do not consider the cost of correction associated with a sentence nor the cost associated with single BOWs as we assume a uniform cost. When implementing a cost-sensitive retraining process one should keep in mind that BOWs come from different sources that are linked to different costs.

We only investigated the effect of retraining a single POS tagger. Another way to improve annotation might be to retrain and use multiple taggers. It was found that when employing a number of different taggers accuracy can be improved [58, 65]. The different POS tagging approaches and algorithms are to some extent complimentary in the errors they make.

We discussed how redundancy in our training set can be reduced, but redundancy is not inherently bad. If we have redundancy in the training sentences selected from the GMB, it might also mean that there is a higher chance that possible errors still persisting in the training set are most likely counterbalanced by a greater number of correct examples. This implies that when trying to eliminate redundancy in the training data, like it is done with approaches like AL,

flaws in the annotation might carry more weight. This might be most apparent when looking at words that are newly added to the training data. Since we add new genres to the training data we suspect that also a number of so far unseen words are added to the training data. If, by inaccurate correction, a newly added word is incorrectly tagged and added to the training data, this error is likely to propagate into the tagging process.

- (4) Pray help me now and scold me afterwards .  
 NN NN PRP RB CC VB PRP RB .

Due to carelessness the word *Pray* in example (4) might not be spotted as incorrectly tagged. The tagger has seen this word in the training data and will be more confident with assigning the label to the word *Pray*. This means that sentences containing that word will less likely be chosen as suitable for correction and retraining by algorithms in AL such as QBU. Since QBC splits the training data among its committee members, it might be less prone to such an error.

One of the biggest objections against the validity of the results is the composition of the silver standard test set. The silver standard cannot guarantee a representative estimate since the initial set of BOWs covers an arbitrary part of the data. For adding more BOWs only a method building on the disagreement of two taggers was used. This could have been augmented by other methods specifically developed for detecting errors or inconsistency in annotation to ensure an extensive coverage. Despite those shortcomings, we think that the silver standard actually provides fairly good coverage.

Along the lines of the described deficiency of the silver standard to give a global estimate of the performance of the tagger is the problem of measuring newly introduced errors. As stated above there is a small percentage of tags that are changed in the GMB test set without being covered by a BOW. However on the basis of the design of our silver standard we can assume that we are still able to estimate the rate of newly introduced errors. The tagger that was used to find disagreements in the creation process of the silver standard, was trained in a similar way (namely: adding sentences containing BOWs to the training data) to the taggers that were tested later on the data set. This allows to assume that a large quantity of changes that are likely to appear by this means of retraining were manually checked and added as test material. This assumption is supported by the fact that only a small percentage of tokens not covered by a BOW was changed. We do not want to disclaim that employing a more sophisticated method of detecting errors would improve coverage even further. However, we expect that this significantly increases the amount of manual effort needed. The only definite way to a reliable estimation of error rate would be to turn the silver standard into a gold standard by full manual annotation.

## 6. Pilot Study: Building a smarter tagger

As we have seen from the results of the retraining process presented in chapter 5, the accuracy of a tagger can be improved by adding automatically tagged and manually corrected sentences to its training data. The tagger learns, in a way, from the mistakes it has made earlier. Those very mistakes, that had been corrected and the confirmed/correct tagging has been learned, are consequently less likely to appear again in the future. This improvement, however, is only possible if the tagger has access to the corrections in the training process. Even after retraining it is not guaranteed that a sentence, that was corrected and used in retraining, gets tagged correctly. The tagger might still make the same mistake as before or introduce new errors. This is due to the fact that the tagger’s model abstracts, which is a desired and vital characteristic of a good tagger.

In the general design, a tagger uses only tokens as an input and is therefore ignorant of the corrections (*known labels*), that might be associated with the tokens, in the process of tagging. This is in fact an unnecessary restraint. The tagger has to ‘guess’ while the true labeling is already known, and if the tagger ‘guessed’ incorrectly, its decision gets corrected not before the tagging is finished. We hypothesize that by making fractional reliable tagging information available to the tagger, its accuracy can be increased. We believe the tagger will benefit in one major way: The *known labels* will positively influence the decision making for the so far unknown labels. In tagging contextual information and the fitting together of all tags in a sequence play an important role. By effectively fixing the tags for a subset of the sequence, the contextual information for the rest changes and the tagging can be improved. In general, this approach builds on the idea of reducing the ambiguity in the tagging process.

As outlined above, such a tagger might be helpful in environments where only patchy annotation is available. For example, employing non-expert annotators can result in incomplete annotation. The non-experts might not be able to annotate a full sentence (i.e. decide all cases) or are not presented with the full set of choices, for example in a GWAP where they only have to decide about a single problem (e.g. the correct label for a single word). Of course, a situation, like it is present in the GMB, where corrections get provided on token level, fits the scope of a possible application perfectly.

The aim of this chapter is to present an enhancement to the general POS tagger paradigm. In a proof-of-concept scenario we want to test, whether it provides appreciable improvement and we try to get an impression of its effects and possibilities.

### 6.1 Method

The proposed enhancement to the tagger can be implemented quite easily. Two steps are necessary: The *known labels* have to be made accessible to the tagger at run time and the tagger must definitely choose the *known label* for a token if available. While the former is trivial, the latter can be achieved by simply

allowing only one tag for the target token. This can be done by effectively setting the probability for this tag to 100% (e.g. by simulating a dictionary look-up) and thus reducing the ambiguity to zero.

For the experiment, an implementation of an ME tagger was used that exhibits all characteristics of a state-of-the-art POS tagger [33]. It was merely modified to allow the use of true labels in the tagging process as indicated above. Sections 02 – 21 of the WSJ part of the CCGbank serve as training data. The tagging algorithm and the chosen feature set is a reimplementaion of the MXPOST [42] and achieves an accuracy of over 96% when tested on test sections (22 – 24) of the WSJ corpus of the CCGbank. We give a formal description of this tagging model in chapter 4.

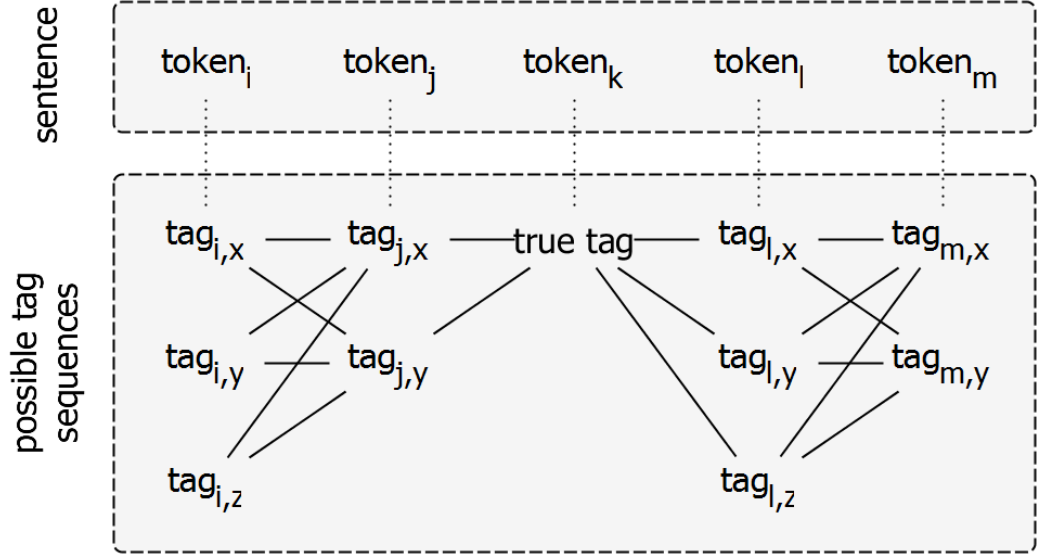


Figure 6.1: Schema of applying a known label in the tagging process.

Since this tagger model, similar to most probabilistic taggers, includes surrounding tags of the target word as features in the decision process when calculating the most likely tag sequence for a sequence of tokens, we expect the effects of the modifications to be visible in the direct context of the *known tags*. Figure 6.1 depicts how a fixed tag influences the tagging of its direct context.

In the ME tagging model the probability of a sequence of tags  $y_1 \dots y_n$  given a sequence of words  $w_1 \dots w_n$  is approximated in following form:

$$p(y_1 \dots y_n | w_1 \dots w_n) \approx \prod_{i=1}^n p(y_i | x_i) \quad (6.1)$$

where  $x_i$  denotes the context for a word  $w_i$ . As described above, in our enhancement we want to fix the probability for a tag  $y_j$  (where  $1 \leq j \leq n$ ) to 1 if we know it is a *true tag*.

For evaluation we compare the output of two taggers (or rather two tagger configurations): A tagger in standard configuration and our proposed augmented tagger, that makes use of the *true labels* at tagging time. Both taggers use the same model, i.e. both taggers are trained in the same way using the same training data. Since for this pilot study we are only interested in changes that

do not appear on the pre-labeled tokens, the output of the standard tagger is corrected using known true labels. Differences in tagging, as depicted in figure 6.2, are then evaluated qualitatively. We are interested in the number of positive changes made by the enhancement, as well as in the kind of tag, in which tag differences appear most frequently. We will focus on disagreement pairs where correct solutions can easily be determined and avoid those tokens that can have multiple allowed tags. An example for a token with more than one possible interpretation is given in (5) (taken from [49]). The example sentence has two different readings depending on whether ‘*Sampling*’ is interpreted as a noun or as a participle of the corresponding verb.

- (5) Sampling/NN|VBG data can be fun.

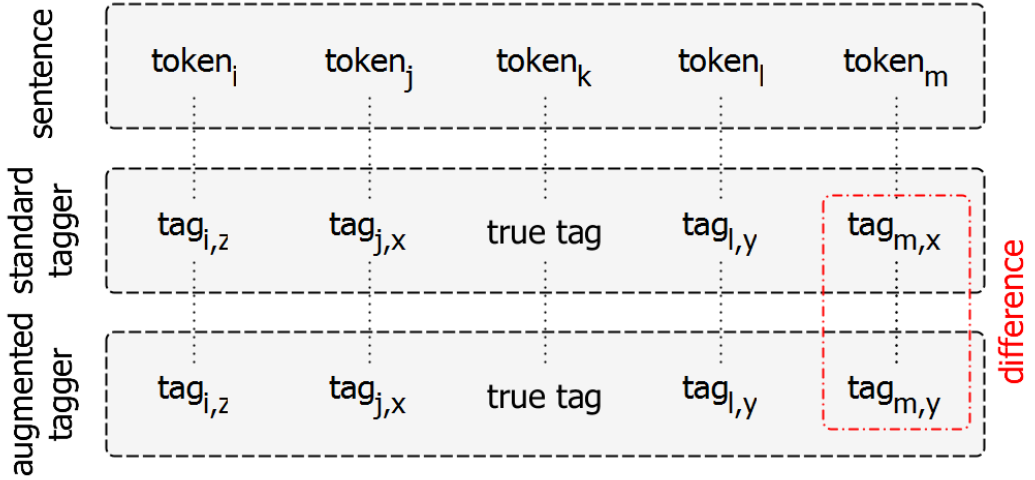


Figure 6.2: Schema of the expected difference in output of the two taggers.

The evaluation data set consists of all sentences that contain at least one manual correction out of the entire GMB. Since quotation marks are not included in the training data, we also exclude all sentences with quotations marks from the test set. This results in a test set with 12,015 sentences and 285,359 tokens. There are 15,152 BOWs, i.e. known labels, in these data.

Since tagging is a deterministic procedure, all differences between the two taggers are due to the fixed incorporated *known labels*. This implies that changes will only appear in sentences where the tagger actually mislabels tokens that have an associated *known label*. For 27.17% of the tokens with known tags the standard tagger already produces the true label. Thus we do not expect any changes in analysis in the context of these tokens.

## 6.2 Results

We observed 619 differing tags in 568 sentences. This shows that changes are only triggered by a small subset of BOWs and consequently only appear on a small percentage (4.7%) of sentences.

As expected, most changes appear in the direct context (2-token range) of the pre-labeled tokens. In a few cases, changes also appear outside of the direct

context. These changes always occur together with changes within the direct context. The altered analysis that is started by the fixed label is propagated in a chain-like fashion.

Table 6.1 gives an overview of the disagreements in tagging. For brevity only labels appearing in the top 10 most common changes are included. The majority of the changes appear on a few tags (e.g. NNP, VBD, ...) only.

*Table 6.1: Selection/Digest of differences in tagging between the improved and the default tagger.*

	Standard Tagger							
		JJ	NN	NNP	NNS	VBD	VBN	VBZ
	Augmented Tagger							
	JJ	–	27	56	0	4	2	0
	NN	16	–	88	3	1	0	0
	NNP	15	33	–	4	0	0	0
	NNS	1	6	28	–	0	0	21
	VBD	5	1	0	0	–	29	0
	VBN	1	0	1	1	60	–	0
	VBZ	0	0	1	4	0	0	–

It is easy to spot that some pairings of differences are distinctly more often than others. Table 6.2 gives the result of the evaluation of four different disagreement pairs. For each disagreement pair (*tag1* (chosen by the default tagger)  $\rightarrow$  *tag2* (chosen by the improved tagger)), we table the number of correct and incorrect assigned tags by the improved tagger for both possible combinations. Additionally the combined accuracy for a pair of tags is given. The evaluation gives two pairs with a decent performance (NNS  $\leftrightarrow$  NNP, VBZ  $\leftrightarrow$  NNS) and two pairs with rather poor performance (NNS  $\leftrightarrow$  NNP, VBD  $\leftrightarrow$  VBN). In the latter two cases the poor results seem to be unidirectional. For example, results are reasonably good for changes from VBN to VBD, but unsatisfactory for the other direction.

By investigating the sentences in which a proper noun tag was wrongly changed to a noun tag (NNP  $\rightarrow$  NN), we found that 57 times the word ‘*Baghdad*’ was the target token that was incorrectly classified as a noun. Example (6) shows a typical misclassification for this case.<sup>1</sup> When checking the tagger model we found, that this token was not present in the training data. Due to the fact that it is an unknown word it has high uncertainty. The changes in the tagger seem to have worsened the performance on this token.

- (6) North of Baghdad , gunmen killed a policeman .  
**RB** IN NNP/NN , NNS VBD DT NN .

For many of the misclassifications in the case of VBD  $\rightarrow$  VBN, a certain pattern was identified in which the misclassification makes sense. The wrongly tagged verb was often preceded by a noun in the plural or singular form, as shown

<sup>1</sup> The fixed label is set in **bold face**.

Table 6.2: Evaluation on interesting cases of newly assigned tags by the augmented tagger depending on the tagging of the standard tagger.

standard $\rightarrow$ augmented	correct	false	combined accuracy
NN $\rightarrow$ NNP NNP $\rightarrow$ NN	33 10	0 78	35.54 %
NNS $\rightarrow$ NNP NNP $\rightarrow$ NNS	2 21	2 7	71.88 %
VBD $\rightarrow$ VBN VBN $\rightarrow$ VBD	4 22	56 7	29.21 %
VBZ $\rightarrow$ NNS NNS $\rightarrow$ VBZ	17 2	4 2	76 %

in example (7). A correct form of appearance for this pattern is given in example sentence (8).

(7) A Lion used to prowl about ...  
DT **NN** ~~VBD~~/VBN TO VB IN

(8) Major exports made up of copra and ...  
JJ NNS **VBN** RP IN NN CC

Next to the shortcomings there is a number of positive changes. These are especially prevalent in the pairs NNS  $\leftrightarrow$  NNP and VBZ  $\leftrightarrow$  NNS. For both pairs only one direction gives meaningful results as the other is underrepresented. Instances of such positive changes for each pair respectively are presented in examples (9) and (10).

(9) Dozens of Egyptians protested ...  
NNP/NNS IN **NNS** VBD

(10) His early blues hits included ...  
PRP\$ JJ **NN** ~~VBZ~~/NNS VBD

## 6.3 Discussion

We can draw two major conclusions from the evaluation of the tagger: Only a small number of changes is produced by the proposed method and the changes give only moderate improvement. Our evaluation covers a large quantity of the changes but might not be fully representative in terms of accuracy as not all cases are covered. A large-scale evaluation on a gold-standard data set would be a proper way to estimate the rate of improvement such an enhancement yields to a tagger. However this raises the problem how to realistically model incomplete annotation on the input data. The evaluation suggests that the results depend on the type of tag that is fixed for the tagging. This means that performance may vary with the distribution of fixed tags in the data. Additionally, it is interesting to see that most of the changes appear on the typical (for a tagger) hard-to-distinguish cases. Those cases include ambiguous words (verb-noun homographs)

as well as grammatical distinctions (past tense vs. past participle). Cases of uncommon distinctions might also be well justified but are infrequent in the results.

Two weak points that gave rise to poor performance could also be identified. The results showed that the augmented tagger systematically misclassified a number of tokens as a past participle. It also performed remarkably poor on one particular token that was not present in the training data. A linguistic prototype that matches the pattern of the misclassification could be found to describe the former problem. The poor results on the latter case could be mitigated if the tagger is combined with suitable retraining. Since we only assess the correctness of the labels produced by the augmented tagger, we cannot draw any conclusions about the correctness of tagging without augmentation. This means, if the label applied by the improved tagger was incorrect, it does not imply that the tag produced by the standard tagger was correct. However, this might help to identify difficult cases.

There are two possible main explanations for the fact, that there were only a few changes produced: On the one hand it could simply mean that the method we propose has only little effect. The modification applied to the tagger might not be strong enough to force a supposedly correct analysis. On the other hand it is possible that there are not many errors that could be found in our data. Building on the hypothesis that if a human annotator corrects a label in the data, s/he will also correct surrounding labels, we can assume that not many errors might be present in the direct context of corrected labels. When taking into account that changes only appear in the direct context of the fixed labels, this might be an explanation for the small number of changes.

In addition to the conclusions drawn from the results, we want to discuss the usability and possible scope of application of our tagger. Our enhancement might only be useful in a small number of contexts as incomplete annotation is not very common. In the traditional corpus annotation paradigm sentences always get labeled in their entirety. Settings similar to the GMB, where annotation is replaced by correction, are most suitable. An important prerequisite is that the provided fixed labels are reliable.

When using the presented enhancement to the tagging, information about the tagger’s uncertainty on a sequence containing a known label is altered or lost. Detecting possible errors in annotation or cases that are hard for the tagger might become more difficult, as they are obscured by the fact that the tagger has less choices available and thus a higher confidence. This implies the tagger is not suitable for approaches like AL or similar.

This tagger might be a way to reduce errors from human annotators that manually correct annotated text. It can indicate errors otherwise missed by an annotator by giving feedback about changing analysis on other tokens directly after the corrections are applied. For this purpose it is a much more convenient way to make ad hoc use of the provided corrections than retraining and tagging, since it is significantly faster.

The tagger that we augmented with an extension for our experiments builds on the same algorithm as the tagger (C&C) that is used in the GMB project. We can expect similar results from augmenting the C&C tagger. Due to different parameter settings (e.g. smoothing), however, results are not fully transfer-

able/comparable. With a retrained tagging model, such as one presented in the previous chapters, only a small number of tokens with associated *known labels* will be incorrectly tagged. Thus augmenting the tagger in this situation would yield hardly any differences.

We only presented an enhancement to a tagger used for POS annotation. However, taggers with an analog design used in other tasks like NE recognition might benefit similarly and offer an interesting avenue for future work. We are currently not aware of any published work on a tagger extension similar to the approach proposed here.

## 7. Conclusion

In this chapter we summarize the most important results of this thesis in order to answer our research questions. Furthermore, we want to advice on how findings of this thesis can be implemented to get a benefit for the Groningen Meaning Bank (GMB) project. Finally we give pointers to future work.

### 7.1 Findings

This thesis has described possible ways how to use manually verified tags, the Bits of Wisdom (BOWs), to effectively improve overall part-of-speech (POS) annotation in the GMB corpus. Our results show in general that by retraining the POS tagger after including sentences with BOWs in the training data the annotation within the GMB corpus can be improved by correcting numerous errors. No change in performance was observable on external gold-standard test sets. Our best model, that uses 96% of the BOW-sentences in retraining, managed to make the correct prediction for almost 70% of manually verified tags in the test set, which is an improvement of roughly 65% over the baseline. Employing the improved POS tagger model as an input to the parser leads to a slightly raised parse coverage.

We investigated how the existing BOWs can be sampled most effectively and found that already 70% of the data can give a performance comparable to the one of our best model. Prioritizing those sentences in retraining with a high ratio of verified annotation proved to yield more effect than random selection. A strong effect of retraining in terms of corrected tags was especially visible after adding only a fraction (up to one third) of the available data. When adding more data the improvement effect is slowed down. Furthermore, most changes in tagging were triggered when only using a smaller set of additional training examples. This leads us to the conclusion that the set of provided corrections in the GMB contains a high degree of redundancy. As a consequence, if retraining is applied iteratively, the process can be directed to only allow sentences with a high ratio of BOWs.

In simulating Active Learning (AL) we found that the sampling methods Query by Uncertainty (QBU) and Query by Committee (QBC) are able to select training data that have a retraining value comparable to data selected by BOW ratio. Selecting sentences with AL enables the tagger to achieve 90% of the performance of our best model while applying almost half the data in comparison to random selection. It is essential, however, that the data selected by AL are reliably tagged. AL suggests itself to be used to guide future annotation effort in the GMB since it provides a way of ensuring an effective impact of the corrections provided.

In addition to the findings of the retraining experiments, we proposed a way of making sporadic corrections in annotation directly available to the tagger without retraining. In a pilot study we showcased how a tagger can use known labels at tagging time to augment its decision making for other tags in the sequence. Testing this extension on the GMB showed only modest results. The main limitation we found was that only very few changes in the annotation were triggered

by the augmented tagger. However, we believe, that in the context of creating a gold-standard corpus it is a promising approach and presents an avenue for further research.

## 7.2 Advice

One of the main findings of this thesis is that retraining (the POS tagger) helps. By continuing to use the standard models, the GMB project would restrain itself unnecessarily. Retraining, even with only a small fraction of the added knowledge of the correction, can help to advance the corpus.

The GMB project should adopt a way how to direct annotation effort. This would help to ensure that corrections are of high informative value and the provided effort is used beneficially. Annotation effort could be directed towards interesting and hard cases to get an optimal effect for example by using AL. But also a simpler procedure like checking differences in tagging after applying an improved tagger model could already be an effective approach. Since it was found that sentences with a high ratio of BOWs are especially helpful in retraining, focusing on further correcting and verifying sentences which have already been corrected at least once can help to enhance the existing information. This would also help to mitigate negative effects of using sentences that carry a BOW in retraining by further checking so far unverified annotation. In addition it would be a first step in creating a gold-standard POS part within the corpus. Our work started the development of a gold-standard part on the POS level in form of a silver standard. By gradually increasing the amount of manually verified tags the value of this part could be increased further.

When choosing AL for future annotation, it might be wise not only to target a single level of annotation but rather multiple. The motivation should be improving annotation on all levels of annotation and pursuing the global goal of the GMB to build a gold standard for semantic representations. One way of fulfilling this intention would be to use *alternating selection* [44]. In this from of AL instances are chosen to maximize training utility on more than one level of annotation. This fits the idea of the GMB since it is focused on many levels of annotation.

## 7.3 Future Work

In this thesis we only evaluated the effect of retraining on the POS level of annotation. But also any other level of annotation with a high amount of BOWs, such as the named entity (NE) level, lends itself to retraining. Investigating retraining on other layers could help to find a more general approach that applies to multiple types of annotation.

In addition to the main evaluation of retraining on the POS level, the effect on other layers of annotation in the pipeline was only tested on the example of the parser. To get a proper estimation of the influence on higher levels of annotation an analysis in more depth is needed. The evaluation could also include how parallel retraining on different levels influences the rest of the corpus.

In retraining the POS tagger we chose to set the training parameters (e.g. smoothing parameter, feature cutoff, ...) to the same values as the default model. However when building and tuning an improved model this parameters should also be adjusted for best results. For example, Curran and Clark [17] suggest that the smoothing parameter is dependent from the number of training examples used. Since the number of training examples largely increases this is one example for possible tuning.

Future work should also include actual periodical retraining and a later evaluation in a practical setting to see how the effect of manual corrections changes over time.

# Bibliography

- [1] Brigham Anderson and Andrew Moore. Active learning for hidden markov models: Objective functions and algorithms. In *Proceedings of the 22nd international conference on Machine learning*, pages 9–16. ACM, 2005.
- [2] Shlomo Argamon-Engelson and Ido Dagan. Committee-Based Sample Selection For Probabilistic Classifiers. *Journal of Artificial Intelligence Research*, 11(335):360, 1999.
- [3] Jason Baldridge and Miles Osborne. Active Learning and the Total Cost of Annotation. In *EMNLP*, pages 9–16, 2004.
- [4] Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The WaCky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226, 2009.
- [5] Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. Developing a large semantically annotated corpus. In *LREC*, volume 12, pages 3196–3200, 2012.
- [6] Valerio Basile, Johan Bos, Kilian Evang, and Noortje Venhuizen. A platform for collaborative semantic annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 92–96. Association for Computational Linguistics, 2012.
- [7] Leonard E Baum. An equality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, 3:1–8, 1972.
- [8] Markus Becker, Ben Hachey, Beatrice Alex, and Claire Grover. Optimising Selective Sampling for Bootstrapping Named Entity Recognition. In *ICML-2005 Workshop on Learning with Multiple Views*, pages 5–11, 2005.
- [9] Johan Bos. Wide-Coverage Semantic Analysis with Boxer. In *Proceedings of the 2008 Conference on Semantics in Text Processing*, pages 277–286. Association for Computational Linguistics, 2008.
- [10] Johan Bos, Valerio Basile, Kilian Evang, Noortje Venhuizen, and Johannes Bjerva. The Groningen Meaning Bank. In Nancy Ide and James Pustejovsky, editors, *The Handbook of Linguistic Annotation*. Springer, Berlin, 2015. To appear.
- [11] John Carroll, Ted Briscoe, and Antonio Sanfilippo. Parser Evaluation: a Survey and a New Proposal. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*, pages 447–454, 1998.

- [12] Jon Chamberlain, Massimo Poesio, and Udo Kruschwitz. Phrase detectives: A web-based collaborative annotation game. In *Proceedings of the International Conference on Semantic Systems (I-Semantics' 08)*, pages 42–49, 2008.
- [13] Jon Chamberlain, Kar en Fort, Udo Kruschwitz, Mathieu Lafourcade, and Massimo Poesio. Using games to create language resources: Successes and limitations of the approach. In *The People’s Web Meets NLP*, pages 3–44. Springer, 2013.
- [14] Stephen Clark and James R Curran. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Computational Linguistics*, 33(4):493–552, 2007.
- [15] Stephen Clark, Julia Hockenmaier, and Mark Steedman. Building Deep Dependency Structures with a Wide-Coverage CCG Parser. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 327–334. Association for Computational Linguistics, 2002.
- [16] Stephen Clark, James R Curran, and Miles Osborne. Bootstrapping POS taggers using unlabelled data. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 49–55. Association for Computational Linguistics, 2003.
- [17] James R Curran and Stephen Clark. Investigating GIS and smoothing for maximum entropy taggers. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 91–98. Association for Computational Linguistics, 2003.
- [18] James R Curran and Stephen Clark. Language independent ner using a maximum entropy tagger. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 164–167. Association for Computational Linguistics, 2003.
- [19] James R Curran, Stephen Clark, and David Vadas. Multi-tagging for lexicalized-grammar parsing. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 697–704. Association for Computational Linguistics, 2006.
- [20] James R Curran, Stephen Clark, and Johan Bos. Linguistically motivated large-scale NLP with C&C and Boxer. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 33–36. Association for Computational Linguistics, 2007.
- [21] Markus Dickinson and W Detmar Meurers. Detecting errors in part-of-speech annotation. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 107–114. Association for Computational Linguistics, 2003.

- [22] Kilian Evang, Valerio Basile, Grzegorz Chrupala, and Johan Bos. Elephant: Sequence Labeling for Word and Sentence Segmentation. In *EMNLP*, pages 1422–1426, 2013.
- [23] Jenny Rose Finkel, Christopher D Manning, and Andrew Y Ng. Solving the problem of cascading errors: Approximate Bayesian inference for linguistic annotation pipelines. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 618–626. Association for Computational Linguistics, 2006.
- [24] Yoav Freund, H Sebastian Seung, Eli Shamir, and Naftali Tishby. Selective sampling using the query by committee algorithm. *Machine learning*, 28 (2-3):133–168, 1997.
- [25] Eugenie Giesbrecht and Stefan Evert. Is Part-of-Speech Tagging a Solved Task? An Evaluation of POS Taggers for the German Web as Corpus. In *Web as Corpus Workshop (WAC5)*, page 27, 2009.
- [26] Ben Hachey, Beatrice Alex, and Markus Becker. Investigating the effects of selective sampling on the annotation task. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 144–151. Association for Computational Linguistics, 2005.
- [27] Robbie Haertel, Eric Ringger, Kevin Seppi, James Carroll, and Peter McClanahan. Assessing the Costs of Sampling Methods in Active Learning for Annotation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers, HLT-Short '08*, pages 65–68, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.
- [28] Robbie A Haertel, Kevin D Seppi, Eric K Ringger, and James L Carroll. Return on investment for active learning. In *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, volume 72, 2008.
- [29] Julia Hockenmaier and Mark Steedman. CCGbank: a corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396, 2007.
- [30] Nancy Ide, Collin Baker, Christiane Fellbaum, and Charles Fillmore. MASC: The manually annotated sub-corpus of American English. In *In Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC)*. Citeseer, 2008.
- [31] Hans Kamp and Uwe Reyle. From Discourse to Logic: an introduction to modeltheoretic semantics, formal logic and Discourse Representation Theory. *Hingham, MA: Kluwer*, 1993.
- [32] Graeme Kennedy. *An introduction to corpus linguistics*. Routledge, 2014.
- [33] Zhang Le. Maximum entropy modeling toolkit for python and c++. *Natural Language Processing Lab, Northeastern University, China*, 2004.

- [34] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of the eleventh international conference on machine learning*, pages 148–156, 1994.
- [35] Hrafn Loftsson. Correcting a POS-tagged corpus using three complementary methods. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 523–531. Association for Computational Linguistics, 2009.
- [36] Christopher D Manning. Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics? In *Computational Linguistics and Intelligent Text Processing*, pages 171–189. Springer, 2011.
- [37] Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, 19(2):313–330, 1993.
- [38] Charles F. Meyer. *English Corpus Linguistics : An Introduction*. Studies in English Language. Cambridge University Press, 2002.
- [39] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer Science & Business Media, 2006.
- [40] Miles Osborne and Jason Baldridge. Ensemble-based Active Learning for Parse Selection. In *HLT-NAACL*, pages 89–96, 2004.
- [41] Slav Petrov, Dipanjan Das, and Ryan T. McDonald. A Universal Part-of-Speech Tagset. *CoRR*, abs/1104.2086, 2011.
- [42] Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, pages 133–142, 1996.
- [43] Dietrich Rebholz-Schuhmann, Antonio José Jimeno Yepes, Erik M Van Mulligen, Ning Kang, Jan Kors, David Milward, Peter Corbett, Ekaterina Buyko, Elena Beisswanger, and Udo Hahn. CALBC silver standard corpus. *Journal of bioinformatics and computational biology*, 8(01):163–179, 2010.
- [44] Roi Reichart, Katrin Tomanek, Udo Hahn, and Ari Rappoport. Multi-Task Active Learning for Linguistic Annotations. In *ACL*, volume 8, pages 861–869, 2008.
- [45] Eric Ringger, Peter McClanahan, Robbie Haertel, George Busby, Marc Carmen, James Carroll, Kevin Seppi, and Deryle Lonsdale. Active learning for part-of-speech tagging: Accelerating corpus annotation. In *Proceedings of the Linguistic Annotation Workshop*, pages 101–108. Association for Computational Linguistics, 2007.
- [46] Eric K Ringger, Marc Carmen, Robbie Haertel, Kevin D Seppi, Deryle Lonsdale, Peter McClanahan, James L Carroll, and Noel Ellison. Assessing the Costs of Machine-Assisted Corpus Annotation through a User Study. In *LREC*, volume 8, pages 3318–3324, 2008.

- [47] Nicholas Roy and Andrew McCallum. Toward Optimal Active Learning through Monte Carlo Estimation of Error Reduction. *ICML, Williamstown*, pages 441–448, 2001.
- [48] Marta Sabou, Kalina Bontcheva, Leon Derczynski, and Arno Scharl. Corpus annotation through crowdsourcing: Towards best practice guidelines. In *Proceedings of LREC*, 2014.
- [49] Beatrice Santorini. Part-of-speech tagging guidelines for the Penn Treebank Project (3rd revision). 1990.
- [50] Sunita Sarawagi. Information Extraction. *Found. Trends databases*, 1(3): 261–377, March 2008.
- [51] Helmut Schmid. Tokenizing and part-of-speech tagging. *Corpus Linguistics. An International Handbook*, 1:527–551, 2008.
- [52] Tyler Schnoebelen and Victor Kuperman. Using Amazon Mechanical Turk for linguistic research: Fast, cheap, easy, and reliable. *Retrieved on April, 1, 2011*.
- [53] Burr Settles. Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6(1):1–114, 2012.
- [54] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1070–1079. Association for Computational Linguistics, 2008.
- [55] H Sebastian Seung, Manfred Oppel, and Haim Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294. ACM, 1992.
- [56] Claude E Shannon. A Mathematical Theory of Communication. *Bell System Tech. J.*, 27:379–423, 1948.
- [57] Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Y Ng. Cheap and fast — but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 254–263. Association for Computational Linguistics, 2008.
- [58] Drahomíra ”johanka” Spoustová, Jan Hajič, Jan Votrubec, Pavel Krbec, and Pavel Květoň. The Best of Two Worlds: Cooperation of Statistical and Rule-based Taggers for Czech. In *Proceedings of the Workshop on Balto-Slavonic Natural Language Processing: Information Extraction and Enabling Technologies*, ACL ’07, pages 67–74, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics.
- [59] Mark Steedman. *The syntactic process*, volume 24. MIT Press, 2000.

- [60] Katrin Tomanek and Udo Hahn. Semi-supervised active learning for sequence labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1039–1047. Association for Computational Linguistics, 2009.
- [61] Katrin Tomanek, Joachim Wermter, and Udo Hahn. An Approach to Text Corpus Construction which Cuts Annotation Costs and Maintains Reusability of Annotated Data. In *EMNLP-CoNLL*, pages 486–495, 2007.
- [62] Nicola Ueffing and Hermann Ney. Using POS information for statistical machine translation into morphologically rich languages. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 347–354. Association for Computational Linguistics, 2003.
- [63] Hans van Halteren. *Syntactic wordclass tagging*. MIT Press, 1999.
- [64] Hans Van Halteren. *The detection of inconsistency in manually tagged text*. na, 2000.
- [65] Hans Van Halteren, Jakub Zavrel, and Walter Daelemans. Improving Accuracy in Word Class Tagging through the Combination of Machine Learning Systems. *Computational linguistics*, 27(2):199–229, 2001.
- [66] Noortje Venhuizen, Valerio Basile, Kilian Evang, and Johan Bos. Gamification for word sense labeling. In *Proc. 10th International Conference on Computational Semantics (IWCS-2013)*, pages 397–403, 2013.
- [67] Atro Voutilainen. Part-of-speech tagging. *The Oxford handbook of computational linguistics*, pages 219–232, 2003.
- [68] Aobo Wang, Cong Duy Vu Hoang, and Min-Yen Kan. Perspectives on Crowdsourcing Annotations for Natural Language Processing. *Language resources and evaluation*, 47(1):9–31, 2013.
- [69] Lars Wissler, Mohammed Almashraee, Dagmar Monett, and Adrian Paschke. The Gold Standard in Corpus Annotation. In *Proc IEEE Germany Student Conference*, 2014.
- [70] Daniel Zeman. Reusable Tagset Conversion Using Tagset Drivers. In *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, pages 213–218, Marrakech, Morocco, 2008. European Language Resources Association.

# List of abbreviations

Notation	Description	Page List
AL	Active Learning	1, 14–18, 24, 25, 28, 31–34, 36, 37, 43, 45, 46
BOW	Bit of Wisdom: A manual correction or a verification of annotation authored by a human annotator	3–5, 8–10, 18–25, 28–37, 40, 45, 46
CCG	combinatory categorial grammar	7–10, 27, 35
DRS	Discourse Representation Structures	7, 10
GMB	Groningen Meaning Bank	3–10, 15, 18, 20–22, 27–31, 34–38, 40, 43, 45, 46
GWAP	game with a purpose	8, 14, 15, 38
HMM	Hidden Markov Model	11, 12, 26
MASC	Manually Annotated Sub-Corpus	22, 28, 36
ME	maximum entropy	9, 10, 12, 19, 39
ML	machine learning	11, 15
NE	named entity	3, 7, 9, 15–17, 44, 46
NLP	Natural Language Processing	3, 7, 11, 14, 15, 17, 26
POS	part-of-speech	1, 3, 5–13, 15–22, 24–28, 34–36, 38, 39, 44–47
QBC	Query by Committee	24, 26, 31–33, 37, 45
QBU	Query by Uncertainty	24, 25, 31–33, 37, 45
TE	token entropy	25
VE	vote entropy	26
WSJ	Wall Street Journal	17, 20–22, 27, 28, 34, 36, 39

# A. Part-of-speech Tags

Tag	Short Explanation
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LQU	Opening quotation mark
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP\$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
RQU	Closing quotation mark
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VCN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP\$	Possessive wh-pronoun
WRB	Wh-adverb
#	Pound sign
\$	Dollar sign
.	Sentence-final punctuation
,	Comma
:	Colon, Semi-colon