Evaluating the Performance of Automated Part-of-Speech Taggers on an L2 Corpus

Craig Hagerman

第二言語コーパスの自然言語形態素解析に対する評価

ヘガマン・クレッグ

Abstract

Automated Part-of-Speech (POS) tagging is commonly on corpora in order to allow for the systematic study. POS tagging is also a fundamental stage in most natural language processing (NLP) tasks. Although there is a long history of research into automated POS tagging in the field of NLP, the vast majority of the research has been on first language texts. Increasingly second language learner corpora are being compiled. As well, increasing use of English as a second language makes the processing of non-native English texts increasingly likely for NLP applications. However, there is very little research into how second language texts affect the performance of automated POS taggers. This paper describes a study which (1) compares the performance of three taggers on native and second language texts and (2) identifies which POS tagger has the highest level of accuracy when faced with second language writing.

Key words: part-of-speech, tagging, corpus, natural language processing

(Received October 1, 2011)

抄　　録

コンピューターによる自動形態素解析は自然言語処理研究において非常に基本的なステージであり、コーパスを使い研究を進める上で必要な要素である。 自動形態素解析の研究は 自然言語処理の分野で長い歴史があるにもかかわらず、そのほとんどは第一言語を基本としたテキストでなされている。英語を第二言語として使われることが多くなっている現在ではコーパスでも英語を母国語としない人々が書いた文面が多く見られる。しかしながら第二言語で書かれた文面が自動形態素解析にどのような影響を与えるかという研究はあまりなされていない。

この論文は３つの形態素解析を使用し第一言語と第二言語のテキストを比べ、どの形態素解析が一番高いレベルで 第二言語のテキストを正確に解析できるかを述べたものである。

キーワード：品詞、タグ付け、コーパス、自然言語処理

(2011 年 10 月 1 日受理)

Corpora are digital collections of spoken or written texts. In order to do quantitative analysis on corpora it is useful to have some form of linguistic annotation, such as part-of-speech (POS) tagging for each token. In the past half century the development and refinement of automated tagging software has allowed this task to be done by computers. This makes the annotation of corpora several orders of magnitude faster. There has been a great deal of research in into POS tagging and currently state-of-the art POS taggers have performance which rivals human annotators. However, the vast majority of the research into POS tagging has been done on first language (L1) corpora. That is, collections such as the Brown corpus, Penn Treebank and British National Corpus (BNC) which are compilations of speech and writing produced by native English users.

There are an increasing number of second language (L2) users of English worldwide, producing speech acts and writing, which is not well represented in the literature on automated POS tagging. Additionally, there are an increasing number of corpora of second language learner's English speech or writing. Linguistic annotation is necessary in order to systematically analyze such corpora. Thus, it is important to study how effective modern, state-of-the-art POS taggers are when used with L2 corpora.

The aim of this paper is to evaluate the performance of three POS taggers on an L2 English corpus. For this research, a 5000 word extract of an L2 corpus was first annotated by each of the taggers, and then manually tagged. A similarly sized extract of the Penn Treebank was also annotated by the taggers for to compare performance between L1 and L2 data sets given identical training data. Finally quantitative analysis was done to examine the taggers' performance.

Background

NLP

Natural Language Processing (NLP) is a branch of computational linguistics which is concerned with automated, computer processing of natural language such as speech acts or texts. There are an increasing number of applications that depend NLP ranging from automated telephone information systems, to automated essay scoring, to information search and retrieval, to machine translation, to automated detection of risk indicators in medical reports, and beyond. This is a diverse, and relatively young area of research that will undoubtedly become more important in the future.

NLP programs are typically constructed as a 'pipeline', or series of discrete processing tasks, increasing in scope. In NLP applications that do some analysis on textual input (including speech processing, since it must also be converted into a textual representation), pre-processing tasks are necessary to prepare the input, after which application-specific processing is done. For example, NLP software for scoring an essay and NLP software for classifying a movie review would have the same initial pre-processing steps. Lexical and syntactic analysis would be done by the following steps:

- split the input text into individual sentences
- tokenize (extract individual words, punctuation, etc.)
- do Word Sense Disambiguation (WSD) on tokens
- annotate each token with its appropriate part-of-speech 'tag'
- build a table of n-grams (adjacent tuples of tokens)
- parse sentence (use parse trees based on a training corpus)
- resolve syntactic ambiguity
- build a parse tree

After this, the two NLP applications would diverge with different application-specific processing. (Automated essay scoring could use term-frequency to assign a weighting to each token and compare the cosine similarity of the resulting vectors with essays in a training corpus. The movie review software could use sentiment analysis to identify the balance between positive and negative expressions in a given review.)

Each stage in the pipeline depends on the performance of the previous stage. None are trivial. Even splitting natural language into sentences is a non-trivial task, but extremely accurate algorithms have been discovered to make this a reliable part of the pipeline. At the lowest level most NLP applications depend on a high degree of accuracy from the POS tagger.

Part-Of-Speech Tagging Overview

Annotating corpora was done manually prior to the availability of automated methods. Tagging texts by hand is a very slow and laborious process. Automated part-of-speech tagging was first explored in 1962-1963 by Harris, Klein and Simmons (Brill, 1994) using hand-written rules. Thereafter, stochastic Markov models proved to be more accurate as large corpora became available.

POS tagging is necessary in many NLP applications as well as for linguistic analysis. Brants (2000) notes that "a large number or current language processing systems use a part-of-speech tagger for pre-processing... Furthermore, there is a large interest in part-of-speech tagging for corpus annotation projects" (p. 224). van Rooy and Schäfer (2002) state that POS tagging "is a type of annotation that is fundamental to most other types of processing of linguistic resources" (p. 325).

Early taggers had an accuracy of 60-70%. Charniak (1997) proved that given a large corpus for training a trivial statistical method can yield 90% accuracy. That is, record the frequency distribution for each word and tag in the corpus. To process a separate test corpus simply assign the tag with the highest frequency. (Mathematically: arg max $p(t \mid w)$). Currently, state-of-the-art taggers have above 97% accuracy. (Note, this assumes a large training corpus, so those results are not comparable to the results described in this paper due to the smaller training corpus used.) It should be kept in mind that humans are not 100% accurate at POS tagging. Inter-annotator agreement on POS tagging is about 98%. This provides an upper bound taggers may aim for.

POS tagging assigns a 'tag' to each word in a text. For example a short sentence such as "Book that flight" using the Penn Treebank tagset would be tagged as:
"Book/VB that/DT flight/NN ./. "(Jurafsky & Martin, 2009, p. 167).
where VB is the tag for 'verb, base form', DT is the tag for determiner, NN is the tag for 'noun, singular or mass and '.' is the tag for a period.

Most words in English are unambiguous. Many closed class words such as 'the' and 'this' have only one possible tag. Yet automated tagging is far from a trivial undertaking. Many of the most commonly used words have more than one possible usage, making their part-of-speech ambiguous. 'Book' in the preceding example can be either a verb (VB) or noun (NN). Thus, automated POS tagging is a largely a disambiguation task.

Literature Review

Commercial research

There has been an increased need to perform NLP tasks on L2 texts. One burgeoning area involves automated evaluation of L2 student essays. There are commercial software suites which assess and automatically grade student writing. One of the leaders in this is ETS which uses automated essay evaluation in the writing portion of the TEOFL, SAT, GRE and other exams they offer. They also create the popular Criterion web-based essay evaluation software for use in writing classes. ETS has published many articles in the field of NLP with regards to L2 analysis. ETS researchers are, by a large margin, the largest source of research on NLP of L2 texts. There has been extremely little independent research done in this field by academics not working for ETS or one of the other commercial enterprises and implicit bias criticized by Ericsson and Haswell (2006).

L2 POS tagging research

Most research on automated annotation has been done on native texts (i.e. texts written by first language speakers of the language). There has been far less research done on how well tagging algorithms perform on L2 texts. This has been commented on by several researchers in recent literature. "POS annotation of learner language has not received much attention in the literature" (Diaz-Negrillo, 2010, p. 12). "Linguistic annotation of learner data has received virtually no attention so far" (Meurers & Wunsch, 2010, p. 1). Given that POS tagging is a foundational stage of a NLP pipeline, as well as an avenue for studying language development, there is clearly a need to judge and compare the accuracy of POS tagging algorithms on L2 texts.

Pravec (2002) notes that "there are many types of annotation that can be done on a corpus: POS tagging, semantic tagging, discourse tagging, error tagging and parsing" (p. 97). Much of the research which has examined annotation of second language corpora has been from an applied linguistics perspective, focused on studying language development, interlanguage, or error analysis, rather than from the computational linguistics perspective of native corpora POS annotation. Tono (1999) research POS tagging as a means of uncovering distinctive interlanguage patterns in a learner corpora. Stefano Rastelli (2009) advocates a form of annotation for learner corpora called SLA-tagging. This is described as a non-strict, non-L1-governed annotation system to advance the study of learner interlanguage. Many other articles in the literature similarly approach L2 tagging from a SLA perspective in order to study interlanguage. Diaz-Negrillo (2010) also advocates alternate POS annotation as a means of compiling data for interlanguage research. Meurers and Wunsch (2010) note that an annotated learner corpora can be a valuable source of empirical data to "verify generalizations and support the formulation of new hypotheses" (p. 1). They suggest a layered linguistic encoding to annotate a learner corpora to support applied linguistics research. Izumi (2005) researched how automated tagging can do error analysis by means of a systematized error typology.

van Rooy & Schäfer (2002), (2003) and Ibanez (2011) did some of the only research into the efficacy of POS tagging on an L2 corpus as part of an NLP pipeline. van Rooy and Schäfer (2002) clearly state that they are not doing corpus linguistics research. van Rooy and Schäfer (2002, 2003) did a study comparing the performance of three taggers for tagging a Tswana learner English corpus. They also examined the effect of learner errors on tagger accuracy. Ibanez did a very similar study, evaluating the performance of taggers on an L2 corpus of Japanese learners writing in Spanish. After evaluating tagger performance, she correlated tagger errors with learner errors to determine the impact of learner language on POS precision. Ibanez states that "it is necessary to study how state-of-the-art natural language processing tools such as part-of-speech (POS) taggers can be (re)used in the learner domain" (Ibanez, 2011, p. 214).

Method

Programming Language and Toolkit

For this research, taggers and NLP processing software were written in the Python programming language using the Natural Language ToolKit (NLTK). NLTK is a Python module in active development with a wide range of NLP processing capabilities built in, allowing for rapid development and testing, making it ideal for such a pilot study. Additionally, NLTK contains excepts or complete copies of a large number of corpora, including a 10% extract of the Penn Treebank, used for training the taggers in this study.

Experimental Data

In NLP research, especially POS tagging research, it has been common practice to use the Penn Treebank. The Penn Treebank is an annotated, parsed corpus created at the University of Pennsylvania, consisting of 1 million articles from the *New York Times*. For a pilot study, comparison of performance using the Penn Treebank provides a useful metric to compare results against the wealth of past research using the Treebank. NLTK includes an excerpt of the Treebank which is freely available for non-commercial researchers. This constituted the training set for each of the taggers. A separate split of the Treebank was created for a testing set to determine the performance of the taggers on a similar L1 data set. Since the Penn Treebank corpus freely available with NLTK is only an extract of the full Treebank a randomly selected 95-5 training/testing split was used. The training set is of far greater importance since it will be used to train each of the taggers for testing the L2 corpus. The testing set of the Treebank is used to provide a rough comparative measure of performance.

A further reason for using the Penn Treebank is to use a consistent tagset. Not all corpora use the same tagset. van Rooy (2002) notes that using a tagset with more tags increases the likelihood of tagger errors within similar categories. Thus, the Penn tagset was chosen, rather than the CLAWS7 tagset (137 tags), the TOSCA-ICLE tagset (220) or the full Brown tagset (226), since the latter have more subtle distinctions between tags within syntactic categories.

An extract of the CEEJUS corpus was used for the testing set. CEEJUS (Corpus of English Essays written by Japanese University Students) is part of the CEEAUS project (Corpus of English Essays written by Asian University Students). It consists of 700 short, timed essays written on one of two prompts:

• *It is important for college students to have a part time job.*
• *Smoking should be completely banned at all the restaurants in the country.*

This is the only freely available corpus of second language Japanese writing available. A 5000-word extract consisting of 21 essays was randomly selected for the testing set. This size of extract was chosen for two reasons. A similarly sized testing set has been used in similar studies (Ibanez, 2011). Additionally, because CEEJUS is not annotated it was necessary to manually annotate the testing set; a time-consuming task that would not be practical on a much larger extract.

Metrics

Precision, recall and f-measure are widely used metrics in part-of-speech tagging (Thouesny, 2011, p. 113). These metrics are used to compare the performance of the tagger trials. For POS tagging, they are instantiated as follows:

$$\text{Recall} = \frac{\text{number of correct POS tags in tagged data}}{\text{number of correct POS tags in gold data}}$$

$$\text{Precision} = \frac{\text{number of correct POS tags in tagged data}}{\text{number of total POS tags in tagged data}}$$

$$\text{f-measure} = \frac{(\beta + 1.0)\ (\text{Precision})(\text{Recall})}{\beta\ (\text{Precision}) + (\text{Recall})}$$

Taggers

Automated taggers are trained on an extract of an annotated corpus (the training set) and the assessed based on their performance on a separate extract of the same corpus, stripped of annotations (the testing set). The original, pre-tagged version of the testing set is commonly referred to as the gold standard, since the original annotations are the standard the tagger should meet.

Most POS taggers are implemented with either rule-based or stochastic algorithms. (Jurafsky & Martin, 2009). Rule-based taggers use a table of disambiguation rules. This could be a table of hand-written rules, but it is far more accurate, as well as less time-consuming to use machine-learning to identify disambiguation rules. Stochastic taggers, on the other hand, “use a training corpus to compute the probability of a given word having a given tag in a given context” (Jurafsky & Martin, 2009, p. 169). In either case, state-of-the-art tagger algorithms use a training set to ‘train’ the tagger. The testing set is then used to determine its performance.

Three different kinds of POS taggers were used in this study to compare tagging performance of different algorithms. The taggers used were a transformational tagger (Brill algorithm), a classifier-based tagger (using a maximum entropy classifier algorithm), and a Hidden Markov Model (Tags ‘n’ Trigrams algorithm) approach. These three taggers represent the approaches to tagging that have proven to be successful with L1 tagging.

Brill

In 1994 Eric Brill proved that a trainable rule-based tagger could achieve performance comparable to a stochastic tagger. “We have demonstrated that the rule-based approach obtains competitive performance with stochastic taggers on tagging both unknown and known words” (Brill, 1994, p. 5). He further reported that “using the tagger without lexicalized rules, an overall accuracy of 96.3% and an unknown word accuracy of 82.0% is obtained” (1995, p. 560). The Brill transformational tagging algorithm was chosen for the present study because it is a rule-based tagger that has state-of-the-art performance, is easy to implement and has a reasonable runtime.

Brill is a transformation-based learning algorithm with a very uncomplicated procedure. “The general idea is very simple: guess the tag of each word, then go back and fix the mistakes” (Bird, Klein & Loper, 2009). It initially uses annotated training data to compile a list of correction rules. It then makes successive passes through a testing corpus, using those correction rules to identify when bad ‘guesses’ have been made and correct those mistakes. In this way, the algorithm transforms an inferior tagging of a testing set into a superior one. Bird, Klein and Loper use an analogy to explain the procedure:

> Suppose we were painting a tree, with all its details of boughs, branches, twigs and leaves, against a uniform sky-blue background. Instead of painting the tree first and then trying to paint blue in the gaps, it is simpler to paint the whole canvas blue, then “correct” the tree section by over-painting the blue background. In the same fashion, we might paint the trunk a uniform brown before going back to over-paint further details with even finder brushes. Brill tagging uses the same idea: begin with broad brush strokes, and then fix up the details, with successively finer changes (2009).

Classifier-based

Classification involves assigning a class label to an input. In the case of POS tagging, the class label is a POS tag. The classifier used in this study is supervised, in that it uses a training set to identify and extract relevant classification features. When the trained classifier is run on the test set it uses a feature detector to determine the tag with the highest probability for each token. The feature extraction and detection can be done by a variety of algorithms. In the present study Naïve Bayes and Maximum Entropy were used. Perkins claims that a classifier-based tagger “is often the most accurate tagger” (2010, p. 107). The tagger with the greatest accuracy is currently Christopher Manning’s (2011) classifier-based Maximum Entropy Stanford Tagger 2.0 which has an overall accuracy of 97.3%, or 90.8% with unknown words (when trained on the full Penn Treebank).

Tags 'n' Trigrams

TnT (Tags 'n' Trigrams) is another statistical tagger, but with a very different implementation than a classifier-based tagger. It is based on second order Markov Models, and thus has many similarities as well as improvements on HMM (Hidden Markov Model) based taggers. The TnT tagger uses a training set to build a table of frequency distributions of unigrams, bigrams and trigrams. When run on a test set it uses all of the n-gram models to predict the probability of each possible tag for each token. Brants (2002) reported that his TnT implementation of an HMM tagger had an overall accuracy of 96.46%, and 85.86% with unknown words (when trained on the full Penn Treebank). According to the Association of Computational Linguistics, TnT has the second highest performance after Stanford Tagger 2.0 (POS tagging (State of the art), 2011).

Results

For each trial of the taggers the total number of errors was recorded and the precision, recall and f-measure calculated. For a gold standard of comparison, each of the taggers was first run on the testing set of the Penn Treebank extract, giving the results summarized in the table below.

Table 1: Results of testing three taggers on Penn Treebank extract

	Precision	**Recall**	**F-measure**
Brill	0.847	0.857	0.842
Classifier	0.998	0.998	0.998
TnT	0.841	0.847	0.844

These results provide a rough upper bound of performance for comparison with trials on the L2 corpus. Due to the limited size of the Penn Treebank extract used for the testing set, the above results can not be considered to be definitive, but rather a guide to hoped-for performance from the tagging L2 texts.

Bird, Klein and Loper (2009) note that if one simple assigns the most common tag to every word in a corpus about an eighth of the tokens will be tagged correctly. This could be considered the lower bound of performance of any tagger.

The same training set (Treebank) and testing set (CEEJUS) was used with all tagger trials. The initial implementation of each of the taggers was trained on the Penn Treebank training set. Subsequently, each tagger was re-implemented using a variant algorithm. The initial baseline results of the three taggers are summarized in the table below.

Table 2: Results of testing three taggers on CEEJUS extract

	Precision	Recall	F-measure	# errors
Brill	0.842	0.834	0.838	368
Classifier	0.865	0.842	0.853	362
TnT	0.744	0.764	0.754	555

Modifications were made to the implementation of each tagger for subsequent trials. The initial implementation of Brill tagger used a fast tagger trainer, the subsequent implementation instead used multiple transformations resulting in an extremely slow training stage. The initial implementation of the classifier-based tagger used a pre-trained version of MaxEnt modeling. Subsequent implementations instead used a naïve Bayes classifier (v.2), a customized MaxEnt classifier (v.3) and a unigram feature detection classifier (v.4). The initial TnT implementation makes use of a second order Markov model, but does not deal with unknown words. (It merely annotates unknown words as 'UNK'). Subsequent implementations made use of a backoff to deal with unknown words. One version (v.2) simply tagged all unknown words as proper nouns (the most common tag), another used a heuristic search algorithm to predict tags for unknown words (v.3) , and a final used the baseline Brill tagger as a backoff (v.4). The performance of these variations are as follows:

Table 3: Results of testing three taggers (modified) on CEEJUS extract

	Precision	Recall	F-measure	# errors
Brill v.2	0.85	0.84	0.84	362
Classifier v.2	0.83	0.8	0.82	397
Classifier v.3	0.8	0.74	0.77	508
Classifier v.4	0.68	0.76	0.72	751
TnT v.2	0.79	0.81	0.8	468
TnT v.3	0.74	0.76	0.75	555
TnT v.4	0.82	0.85	0.84	426

Conclusion

The results of the initial trials of the taggers on the Penn Treebank (L1) corpus and the L2 corpus show that all three taggers have greater accuracy when annotating a native corpus than second language writing. The Brill tagger had the least decline in performance with the precision decreasing by only 0.5%. However recall showed a decline of 2.3%, and f-measure by 0.4%. The classifier-based tagger has the best performance of all the taggers with 99.9% for precision, recall and f-measure for the small extract of the Treebank it was tested on. When the same tagger was applied to the L2 corpus, performance dropped measurably. Precision decreased by 13.3%, recall by 15.2% and the f-measure score by 14.5%. The TnT tagger also experienced a sharp drop in accuracy when annotating the L2 corpus. Precision declined by 9.7%, recall by 8.3% and f-measure by 9%. Thus, in all trials the taggers were shown to perform worse with second language texts.

The algorithms of each of the baseline taggers were modified and refined to see if such adjustments would have a beneficial effect on performance. When using the Brill algorithm was modified to not use a fast method of tagger training, precision, recall and f-measure all experienced an increase. It should be noted that the so-called 'fast' tagger trainer takes very few minutes to run on this small L2 extract. Indeed, most of the tagger versions detailed here take under 2 or 3 minutes to run. However, the Brill algorithm which did not use the 'fast' trainer takes over one hour to run. This is due to the vast increase in the number of transformation rules sought within the training stage. Consequently, although this algorithm displays an increase in accuracy in many practical applications the tremendous increase in processing time might cancel out the performance improvements.

The baseline classifier-based tagger algorithm was revealed to have the greatest accuracy. Subsequent variations on this algorithm showed a large decrease in precision, recall and f-measure scores. Maximum-entropy based classifiers (such as the Stanford tagger) have been shown to have a high level of performance in many applications. Thus it is not surprising that the baseline classifier, using a MaxEnt approach, would be superior to other variations.

The baseline TnT tagger showed the worst performance of the three taggers in annotating either the L1 or L2 corpora. This can be explained by the fact that the baseline TnT algorithm does not make any attempt to annotate unknown words. Thus, if the baseline TnT algorithm encounters a word in the testing set which did not appear in the training set, it will simply annotate it as 'UNK' (unknown). Rather than failing to annotate in this way, the alternate versions of TnT identify a backoff tagger. Thus, when the algorithm comes upon an unknown word, it will pass off the tagging task to the backoff tagger. Such backoffs can be chained together but there is usually no additional improvement in having more than one or two backoffs. The most common class of lexeme in the corpus is nouns. The simple variation (v.2) which merely tagged all unknown tokens as a proper noun improved performance over the baseline. However, the best results were seen by using another tagger with a high degree of accuracy (in this case Brill) as the backoff. Effectively dealing with unknown words improved precision, recall and f-measure to over 80%.

The chart in Figure 1 graphically shows the performance of the three most accurate taggers:

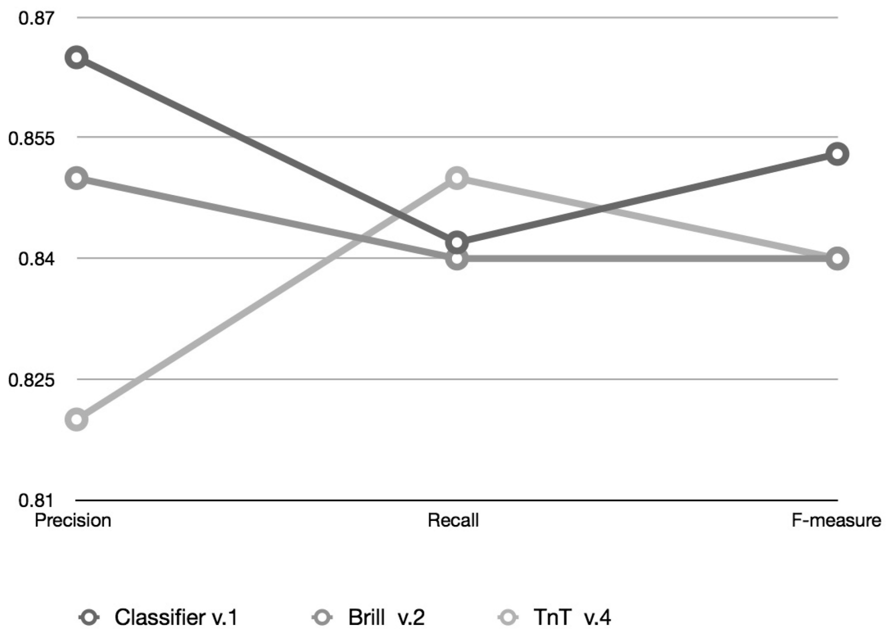

Figure 1. Comparative performance metrics for top three taggers

Of the three taggers analyzed, the classifier-based tagger (baseline / v.1) returned the highest degree of accuracy. Precision and f-measure of the baseline classifier exceed all other taggers and variations. The other taggers with the best performance are Brill (v.2) and TnT (v.4). The recall of the TnT tagger with a Brill tagger backoff, was better than the classifier, but overall within this pilot study the classifier-based tagger shows superior performance.

Future research

As previously mentioned an extract of the Penn Treebank was used for the training set and a manually annotated 5000 word extract of the CEEJUS corpus used for the testing set. This is sufficient for this pilot study to demonstrate that there is a difference in performance when using standard taggers on an L2 corpus, however it is not optimal. There are two potential weaknesses with this approach which must be addressed in future research to determine the decreased performance of an L2 tagger more precisely and guide improved implementations of taggers. Firstly, there is the problem of a genre mis-match. When using machine learning approaches to NLP performance will suffer if the genre of the training and testing sets differ. The Penn Treebank is derived from *New York Times* articles. It is assumed that high level journalism will share some syntactic similarities with academic writing (university student's essays) but it would be optimal to use an annotated corpus of academic writing for the training set. Another weakness with the present research involves the use of two distinct corpora for testing and training. To judge the accuracy of a tagger it is common to use a split (60/40 is typical) of a single, annotated corpus for both training and testing, The present research is concerned with determining how taggers trained on L1 corpora perform when presented with L2 texts (a common occurrence in the wild, which should increase with the increase in English L2 speakers). This makes the use of a native corpus for the training set valid, however, to get a clearer view of the accuracy of a tagger it would be optimal to compare three data sets: training on a native corpus (same genre), training on a training set of an L2 corpus, and testing both versions of the tagger on a testing set from the L2 corpus. It is possible that the International Corpus of Learner English (ICLE) from the University of Louvain may be ideal for this purpose.

References

Association of Computational Linguistics. (2011). *POS tagging (State of the art).* Retrieved from http://aclweb.org/aclwiki/index.php?title=POS_Tagging_(State_of_the_art)

Bird, S., Klein, E. & Loper, E. (2009). *Natural language processing with Python.* Sebastopol, CA: O'Reilly.

Brants, T. (2000). TnT: a statistical part-of-speech tagger ANLC '00 Proceedings of the sixth *Conference on applied natural language processing Association for Computational Linguistics.*

Brill, E. (1994). Some advances in transformation-based part of speech tagging AAAI '94 Proceedings of the twelfth national Conference on artificial intelligence (vol. 1) *American association for artificial intelligence.*

Brill, E. (1995). Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part of Speech Tagging. *Computational Linguistics,* December 1995.

Charniak, E. (1997). Statistical Techniques for Natural Language Parsing. *AI Magazine,* 18(4), 33-44.

Díaz-Negrillo, A., Meurers, D., Valera, S. & Wunsch, H. (2010) Towards interlanguage POS annotation for effective learner corpora in SLA and FLT. In M. M. Jaén & C. P. Basata (Eds.) *Language Forum,* Vol. 36, No 1-2. 139-154.

Ericsson, P. F. & Haswell, R. (Ed.). (2006). *Machine scoring of student essays.* Logan, UT: Utah State University Press.

Ibanez, M. P. V. (2011). An evaluation of part of speech tagging on written second language Spanish. In A. Gelbukh (Ed.) *Lecture notes in computer science*, 6608, 214-226.

Izumi, E., Uchimoto, K., & Isahara, H. (2005). Error Annotation for Corpus of Japanese Learner English. *Proceedings of the Sixth International Workshop on Linguistically Interpreted Corpora* (LINC 2005), 71-80.

Jurafsky, D. & Martin, J. H. (2009). *Speech and language processing* (2nd Ed.). New Jersey: Prentice Hall.

Manning, C. D. (2011). Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics? In A. Gelbukh (Ed.) *Lecture notes in computer science*, 6608, 171-189.

Meurers, D. & Wunsch, H. (2010). Linguistically Annotated Learner Corpora: Aspects of a Layered Linguistic Encoding and Standardized Representation. *Proceedings of Linguistic Evidence 2010,* 218-221.

Perkins, J. (2010). *Python text processing with NLTK 2.0 cookbook*. Birmingham: Packt Publishing.

Pravec, N. A. (2002). Survey of learner corpora. *ICAME Journal*. 26, 81-114.

Rastelli, S. (2009). Learner corpora without error tagging. *Linguistik online*, 38. Retrieved from http://www.linguistik-online.de/38_09/rastelli.html

Thouesny, S. (2009). Increasing the Reliability of a Part-of-Speech Tagging Tool for Use With Learner Language. *Automatic Analysis of Learner Language (AALL'09)*. Retrieved from http://www.sfs.uni-tuebingen.de/~dm/events/aall09/thouesny.pdf

Thouesny, S. (2011). Modeling second language learners' interlanguage and its variability: A computer-based dynamic assessment approach to distinguishing between errors and mistakes. (Doctoral dissertation). Retrieved from: http://icall-research.net/publications/2011_thouesny_PhD_thesis.pdf

Tono, Y. (2000). A corpus-based analysis of interlanguage development: Analysing part-of-speech tag sequences of EFL learner corpora. In Lewandowska-Tomaszczyk, B. & Melia, J. P. (Eds.) *PALC'99: Practical Applications in Language Corpora*, 323-340.

van Rooy, B. & Schäfer, L. (2002). The effect of learner errors on POS tag errors during automatic POS tagging. *Southern African Linguistics and Applied Language Studies,* 20(4), 325-335.

van Rooy, B., & Schäfer, L. (2003). An evaluation of three POS taggers for the tagging of the Tswana Learner English Corpus. In D. Archer, P. Rayson, A. Wilson & T. McEnery (Eds.), *Proceedings of the Corpus Linguistics 2003 Conference Lancaster University*, 16, 835-844.

Voutilainen, A. (2003): Part-of-speech tagging. In: R. Mitkov (Ed.): *The Oxford Handbook of Computational Linguistics* (pp. 219-232). Oxford: Oxford University Press.