Judging Grammaticality: Experiments in Sentence Classification

Joachim Wagner

Jennifer Foster

Josef van Genabith

National Centre for Language Technology
Dublin City University

ABSTRACT

A classifier which is capable of distinguishing a syntactically well formed sentence from a syntactically ill formed one has the potential to be useful in an L2 language-learning context. In this article, we describe a classifier which classifies English sentences as either well formed or ill formed using information gleaned from three different natural language processing techniques. We describe the issues involved in acquiring data to train such a classifier and present experimental results for this classifier on a variety of ill formed sentences. We demonstrate that (a) the combination of information from a variety of linguistic sources is helpful, (b) the trade-off between accuracy on well formed sentences and accuracy on ill formed sentences can be fine tuned by training multiple classifiers in a voting scheme, and (c) the performance of the classifier is varied, with better performance on transcribed spoken sentences produced by less advanced language learners.

KEYWORDS

Grammar Checker, Error Detection, Natural Language Parsing, Probabilistic Grammars, Precision Grammars, Decision Tree Learning, Voting Classifiers, N-gram Models, Learner Corpora

INTRODUCTION

This article is concerned with the task of automatic grammaticality judgments, that is detecting whether or not a sentence contains a grammatical error. As well as being useful for evaluating the output of natural language generation and machine translation systems, automatic grammaticality judgments have applications in computer-assisted language learning. One could envisage, for example, the use of such judgments in automatic essay grading and as a first step towards diagnosing an error and providing appropriate feedback in a language tutoring system. For advanced learners, it might also be helpful to use automatic grammaticality judgments to point learners towards an error without indicating its precise nature.

We describe a method which uses machine learning to exploit three sources of linguistic information in order to arrive at a judgment: part-of-speech n-gram frequencies; the grammaticality judgments provided by a hand-crafted, broad-coverage generative grammar of English; and the output of three probabilistic parsers of English, one trained on *Wall Street Journal* trees, one trained on distorted versions of the original *Wall Street Journal* trees, and the third trained on the union of the original and distorted versions. The first two information sources were described in previous work (Wagner, Foster, & Genabith, 2007). In this article, we demonstrate that incorporating information from probabilistic parsing can lead to significant improvements. In addition, we show how using a series of classifiers in a voting scheme can be used to fine-tune the trade-off between detecting ungrammatical sentences on the one hand and avoiding overflagging on the other hand.

To train our machine-learning-based error detectors, we use a large corpus of well formed sentences and an equally large corpus of ill formed sentences. To obtain the ill formed sentences, we automatically introduce errors into the sentences in the original well formed corpus. Our automatic grammaticality judgments are tested on various types of test data including synthetic ungrammatical sentences (created in the same way as the ungrammatical sentences in the training set), sentences from the International Corpus of Learner English (Granger, 1993) produced by advanced learners of English, and transcribed spoken sentences produced by learners of English at varying levels. Testing the method on the artificially produced ungrammatical sentences allows us to gauge the efficacy of our machine-learning features, while testing the method on real learner data also provides information on potential gaps in our error model.

The article is organized as follows: first, we introduce some basic concepts and situate our work with respect to related research on the problems of automatic grammaticality judgments and error detection. Then, we describe our training and test data and our method for performing automatic grammaticality judgments. This is followed by a discussion of the results of our experiments. Finally, we summarize and provide suggestions for how this research might be fruitfully extended.

BASIC CONCEPTS

We distinguish error detection systems which make use of hand-crafted rules to describe well formed and ill formed structures from purely data-driven systems which use various means to automatically derive an error detector from corpus data. Bender, Flickinger, Oepen, and Baldwin (2004), for example, describe a hand-crafted system in which input sentences are parsed with a broad-coverage generative grammar of English which aims to describe only well formed structures. If a sentence cannot be parsed with this grammar, an attempt is made to parse it with mal-rules which describe particular error types. In the Method section below, we describe how a broad-coverage, hand-crafted generative grammar is integrated into our data-driven method. In this section, however, we focus on data-driven error detection systems and attempt to categorize these methods according to the nature of the task, the type of features or patterns that are automatically extracted from the data, and the type of data used.

The Nature of the Task

Our work is most closely related to that of Andersen (2007), Okanohara and Tsujii (2007), and Sun et al. (2007) since all three are concerned with the task of automatic grammaticality judgments, that is, classifying a sentence as either grammatical or ungrammatical. Other error detection research focuses on identifying and possibly correcting errors of one particular type, for example, errors involving prepositions (Gamon et al., 2008; Tetreault & Chodorow, 2008b; De Felice & Pulman, 2008), determiners (Han, Chodorow, & Leacock, 2006; Gamon et al., 2008; De Felice & Pulman, 2008), and verbs (Lee & Seneff, 2008), as well as real-word spelling errors (Kukich, 1992; Bigert & Knutsson, 2002).

The Nature of the Pattern

There are many possible features of a sentence to which an error detection pattern can refer. If we are to acquire patterns automatically from corpora, we have to restrict their type in order to make the extraction process tractable. Often, automatically acquired error patterns are limited to word or part of speech (POS) sequences of a certain length (Golding & Schabes, 1996; Verberne, 2002). For example, the sequence of three POS tags "determiner determiner noun" might indicate an error in English while the sequence "determiner adjective noun" does not. The choice of patterns limits the types of errors that can potentially be detected. In the example above, the POS information does not handle agreement phenomena between determiner and noun. This gap can be filled by extending the POS tag set such that singular and plural determiners and nouns are distinguished. In some work, closed class words are not reduced to their POS tags, effectively augmenting the POS tag set to be as fine-grained as possible for prepositions, pronouns, and so forth.

N-grams of words or POSs are widely used but are not the only type of patterns used in previous work. Sun et al. (2007) extend n-grams to noncontinuous sequential patterns allowing arbitrary gaps between words. In addition, patterns are collected for all n. Sjöbergh (2006) uses sequences of chunk types, for example, "NP-VC-PP." The parse trees returned by a statistical parser are used by Lee and Seneff (2008) to detect verb form errors.

The Nature of the Data

Positive reference data only

Grammatical text is available in vast quantities for many languages, for example, in news, parliamentary proceedings, and free electronic books. The simplest method of using such text corpora for grammatical error detection is to treat every pattern that is not attested in the corpus as an indicator of an error. For POS trigrams, for example, the list of all possible trigrams is manageable and simply needs to be ticked off while reading the corpus sequentially. The trigram table would only need 50^3 = 125,000 bits assuming a POS tag set containing 50 tags. For POS n-grams with higher n, or for other types of patterns, more sophisticated indexing methods have to be employed.

The use of only positive reference data will detect more errors (but also reject more grammatical sentences) if more than one occurrence in the reference data is required for a pattern to be acceptable. This stricter criterion can be necessary for various reasons: first, the reference corpus may in fact contain ungrammatical language; second, there may be patterns that sometimes occur in grammatical sentences but are more likely to be caused by an error; and third, tagging errors can distort the reference corpus. These problems grow with the size of the reference corpus and can be counteracted by a higher frequency threshold for acceptable patterns. On the other hand, patterns that are unattested in the reference corpus can still occur in correct sentences. Therefore, it is desirable to generalize from the observed patterns. Bigert and Knutsson (2002) introduce a similarity measure on POS n-grams that extends the set of acceptable n-grams. Gamon et al.(2008), Tetreault and Chodorow (2008b), and De Felice and Pulman (2008) exploit machine learning by using word, POS, and parser features to learn a model of correct usage from positive data only and then compare actual usage to the learned model.

While all these methods only use positive reference data, it should be noted that a small amount of negative data (i.e., an error corpus) is necessary to tune the system parameters (e.g., type of patterns, frequency thresholds, etc.) before testing the final system on unseen test data.

Adding negative reference data

Negative reference data consists of a corpus of (mostly) ungrammatical sentences, optionally annotated with the location and type of errors. If there is error annotation in the negative reference data, patterns indicative of errors can be extracted more reliably. The presence of a pattern in negative reference data reinforces the information gained from the absence of the same pattern in positive reference data. A basic method therefore simply flags all patterns as ungrammatical that appear in the negative data but do not in the positive data. As with the positive data, this method can be extended by looking at the frequencies of patterns. The frequency ratio between positive and negative reference data is a possible measure of the discriminativeness of a pattern (Sun et al., 2007).

As in the case of using only positive reference data, it is possible to generalize to patterns that cannot be found in any of the positive and negative reference data sets. Handcrafted similarity measures are not used here, to our knowledge. Machine-learning methods are applied to automatically induce a classifier that discriminates between grammatical and ungrammatical patterns based on some features of the pattern (e.g., see Andersen, 2007; Okanohara & Tsujii, 2007; Sun et al., 2007) and our previous work (Wagner et al., 2007).

DATA

In this section we describe the data used to train and test our grammaticality classifier. The positive training data consists of sentences taken from the British National Corpus (BNC) (Burnard, 2000). The negative training data is artificially generated by automatically distorting BNC sentences. In order to ensure that this distortion process is realistic, it has been designed to replicate the errors found in a corpus of ungrammatical sentences. Our primary motivation for using artificial error data is that our classifier requires tens of thousands of ungrammatical sentences as training data, and we do not have a suitably large corpus of naturally occurring erroneous data at our disposal. The use of artificial error data is not new: Bigert (2004) and Bigert, Sjöbergh, Knutsson, & Sahlgren (2005), for example, automatically introduce spelling errors into texts and use these in spelling error detection and parser robustness evaluation. Okanohara and Tsujii (2007) generate ill formed sentences (they use the term "pseudo-negative examples") using an n-gram language model and then train a discriminative language model to tell the difference between these pseudo-negative examples and well formed sentences. Smith and Eisner (2005a, 2005b) automatically generate ill formed sentences by transposing or removing words within well formed sentences. These ill formed sentences are employed in an unsupervised learning technique called contrastive estimation which is used for POS tagging and dependency grammar induction. In the following section, we describe our automatic error creation process.

To test our classifier, both artificial and naturally occurring test data are employed. The artificial test data are created in the same way as the training data. The remaining test data consist of sentences taken from various learner corpora and a small held-out set taken from the corpus of naturally occurring errors which is used to inform the automatic error creation procedure. The test data are described in more detail below.

Automatic Error Insertion

We create negative data for our classifier by using an automatic error creation procedure which accepts as input a POS-tagged sentence and outputs a deviant version of the input sentence. The automatic error creation procedure is informed by an error analysis carried out on the sentences in the corpus of English language grammatical errors collected by Foster (2005). The 923 ungrammatical sentences in this error corpus are taken mainly from academic papers, emails, newspaper articles, and website forums. The sentences were corrected in context, resulting in a parallel corpus. The errors in the ungrammatical sentences were then analyzed in terms of the substitute/insert/delete operations which were applied to correct them.

For each input sentence, the error creation procedure attempts to produce five kinds of ungrammatical sentence, each exhibiting a different grammatical error. The five kinds of grammatical error involve a missing word, an extra word, verb form, agreement, and real-word spelling. These error types were chosen because they are the five most frequent error types in Foster's error corpus. The error corpus is fundamentally different from a learner corpus because, although it contains competence errors which occur due to a lack of knowledge of a particular structure, many of the errors are in fact performance slips. Some error types are particularly associated with performance slips (e.g., real-word spelling errors). For other error types (e.g., missing word errors), the error can be a result of language transfer from the writer's mother tongue (***I am psychologist***) or can be a mistake produced because the writer is in a hurry or distracted (*I'm not sure what I'm* ***up tomorrow***). Table 1 shows examples of sentences taken from the error corpus for each of the five error types used in the automatic error creation procedure.

Table 1
Sentences from the Foster Error Corpus (Foster, 2005)

Error type	Example
Missing word	*I'm not sure what I'm* ***up tomorrow****.*
	I am psychologist*.*
Extra word	*Why* ***is do*** *they appear on this particular section?*
	Is our youth really ***in in*** *such a state of disrepair?*
Real-word spelling	*Yoga brings peace and vitality to* ***you*** *life.*
	We can order ***then*** *directly from the web.*
Agreement	*I* ***awaits*** *your response.*
	The first of ***these scientist*** *begin in January.*
Verb form	*Brent would often* ***became*** *stunned by resentment.*
	I having *mostly been moving flat.*

Foster's error corpus also contains instances of covert errors (James, 1998) or errors which result in structurally well formed sentences with interpretations different from the intended ones. An example is the sentence *We can order* ***then*** *directly from the web*. Because the errors in the corpus were observed in their discourse context, it was clear that a real-word spelling error had been produced and that the intended sentence was, in fact, *We can order* ***them*** *directly from the web*. Obviously, these kinds of sentences will pose a particular problem for our classifier which accepts sentences in isolation. A similar point is made by Andersen (2007).[1]

For each error type, the error creation procedure is briefly described below (for a more detailed description, see Foster, 2007).

Missing word errors

The automatic error creation procedure creates missing word errors by deleting a word from a sentence. The likelihood of a word being deleted will be determined by its POS tag. In Foster's error corpus, 98% of the missing word errors involve the omission of the following POSs (ordered by decreasing frequency):

1. determiner (28%)
2. verb (23%)
3. preposition (21%)
4. pronoun (10%)
5. noun (7%)
6. infinitival marker "to" (7%)
7. conjunction (2%)

Adjectives and adverbs are not deleted by the procedure because their omission is likely to result in a well formed sentence.[2]

Extra word errors

Approximately two thirds of the extra word errors are created by duplicating a randomly selected token in the input sentence or by inserting a word directly after another word with the same POS tag (*Why **is do** they appear in this section?*). Adjectives are the only exception because their duplication will tend not to result in an ungrammatical structure (*the long long road*). The remaining extra word errors are created by inserting a random token at a random point in the input sentence.

Real-word spelling errors

A list of real-word spelling errors involving commonly occurring function words (prepositions, auxiliary verbs, and pronouns) is used to insert errors of this type.

Agreement and verb form errors

Agreement and verb form errors are created by searching the input sentence for likely candidates, randomly selecting one of them, and then replacing it with its opposite number form or a different verb form.

Test Data

Some of the test data used to evaluate our classifier are the artificial data created by using the automatic error creation procedure described above. However, we also test the classifier on the following data:

1. essays produced by advanced learners of English (608 sentences) (Granger, 1993; Horváth, 1999; PELCRA: Polish and English Language Corpora for Research and Applications, 2004),

2. transcribed spoken language produced by learners of English at all levels (4,602 sentences),[3]
3. sentences containing mass noun errors produced by Chinese learners of English and a corrected version of these sentences (123 X 2 sentences) (Brockett, Dolan, & Gamon, 2006),[4] and
4. held-out data from the Foster parallel error corpus (44 X 2 sentences)

Advanced learner essays

These essays were produced by advanced learners of English with Hungarian, Polish, Bulgarian, or Czech as their mother tongue. One annotator read through the essays and attempted to judge each sentence as either grammatical or ungrammatical. The grammaticality judgment task is not straightforward for native speakers and has high levels of interannotator disagreement (Snow & Meijer, 1976; James, 1998; Tetreault & Chodorow, 2008a). Because of the unreliability of grammaticality judgment and because only one annotator was available for our experiment, we excluded from our test set those sentences for which the annotator was not confident in her judgment. These "questionable" sentences are often syntactically well formed but contain words which would not be used by a native speaker in the same context and hence would likely be corrected by a language teacher such as *I became* ***even devoted*** *to the British* and *The* ***very first look*** *of the streets shows something else*.

Spoken language corpus

This corpus contains transcribed spoken sentences which were produced by learners of English at all levels (beginner, low-intermediate, intermediate, advanced). The speakers' L1s come from the following set: Amharic, Arabic, Cantonese, French, Icelandic, Indonesian, Italian, Japanese, Korean, Mandarin, Portuguese, Russian, Spanish, Thai, Ukrainian, and Vietnamese. The sentences were produced in a classroom setting and transcribed by the teacher, and the transcriptions were verified by the students.

One annotator examined a 499-sentence subset of this corpus, correcting the sentences to produce grammatical data. Fifty-six of these 499 sentences were found to be grammatically well formed (either covert errors or questionable). Of the remaining 443 sentences which were corrected, 253 contained more than one grammatical error. The 190 sentences containing just one error were classified according to the manner in which they were corrected (insert/delete/substitute): 23 sentences contain an extra word (the most common of which is a preposition); 39 sentences contain a missing word error, with almost half of these being missing determiners; and 66 sentences were corrected by substituting one word for another, with agreement errors as the most common subtype. The remaining 62 sentences contain errors which are corrected by applying more than one correction; for example, the sentence *It is one of* ***reason*** *I became interested in English* was corrected by changing the number of the noun *reason* and inserting the preposition *of* before the noun.

Mass noun error corpus

The sentences in this corpus were found on the internet and were produced by Chinese learners of English. Each sentence contains an error involving a mass noun, for example, *I learnt* ***a few*** *knowledge about the Internet*. Brockett et al. (2006) corrected the sentences, resulting in a parallel corpus.

Held-out data from the Foster error corpus

This is a very small parallel corpus which was collected after completion of the error corpus which we use as a basis for our artificial error creation procedure (see above and Foster, 2005). This corpus was compiled in the same way as the main corpus (i.e., the sentences were encountered while reading, they were corrected in context, and their correction was used to classify the error).

METHOD

In this section, we describe our method for classifying a sentence as grammatical or ungrammatical. We first present our previous work in this area (Wagner et al., 2007) and then describe two extensions to this work: the incorporation of probabilistic parser features and the use of classifier voting. We introduce short names in **bold** for each method that we will use in the figures and the discussion.

N-Gram and XLE Methods

Our previous work (Wagner et al., 2007) compares two simple but fundamentally different methods: a shallow method based on POS n-gram frequencies and a deep method employing a hand-crafted precision grammar. The **ngram** method trains decision trees[5] on feature vectors containing 6 features: the frequency of the least frequent n-gram of the sentence for n = 2, …, 7. The **xle** method parses the input with the ParGram English grammar (Butt, Dyvik, King, Masuichi, & Rohrer, 2002) using the XLE parser engine (Maxwell & Kaplan, 1996). We extract 6 features from the parser statistics output, the most important feature being whether the sentence could be parsed with the grammar without the use of special XLE robustness techniques. Other features include the duration of parsing and the number of possible trees. The method that results from merging the **xle** and **ngram** feature sets will be called **comb**.

Probabilistic Grammar Method

Following Foster (Foster, 2007; Foster, Wagner, & Genabith, 2008), we apply the automatic error insertion procedure to the sentences of the Penn Treebank (Marcus et al., 1994), adjust the parse trees accordingly, and induce three probabilistic grammars: one from the original treebank, one from the distorted treebank, and one from the union of the two treebanks. Each sentence is parsed with all three grammars, and features are extracted from the three resulting parse trees. These features include the difference in logarithmic parse probability between the trees and structural differences between the trees measured using various parser evaluation metrics (Black et al., 1991; Sampson & Babarczy, 2003). The Charniak and Johnson (2005) parser is used to produce the parse trees. Following our previous work (Wagner et al., 2007), we train decision trees on feature vectors extracted from the BNC training data. We will refer to this method as **prob**.

Voting over Decision Trees

In addition to proposing a new basic method, we address the issue of the accuracy trade-off between high accuracy on grammatical data (minimal overflagging) and high accuracy on ungrammatical data (few errors missed). In initial experiments we tried to achieve this by varying the density of errors in the training data, but the accuracy trade-off was difficult to control this way. Instead, we train multiple classifiers (decision trees in our case) and have them vote for the final decision. The accuracy trade-off can be tuned by setting the number of votes that are required to flag a sentence as ungrammatical. For example, overflagging will be minimized if sentences are flagged only when all classifiers concordantly judge them as ungrammatical. However, the classifiers must not be identical. They have to disagree on some sentences for the voting to make a difference. Decision trees are particularly suitable because they are unstable, that is, small changes to the training data can result in large changes to the tree (Bauer & Kohavi, 1999; Breiman,1996).

We train 12 decision trees[6] per cross-validation run (in a 10-fold cross-validation setting) on different subsets of the training data,[7] resulting in 120 decision trees that we can combine with the voting scheme when testing on data that have been kept separate from the training setup. This is the case for the evaluation on the authentic test data. However, the evaluation on the synthetic BNC test data can only involve the 12 classifiers of the respective cross-validation run in the voting. We then report average numbers over all 10 runs.

RESULTS

This section presents results for our various test corpora and classifiers. We will first verify Wagner et al.'s (2007) finding that combining features of different methods helps. We then test our best classifier on real learner data.

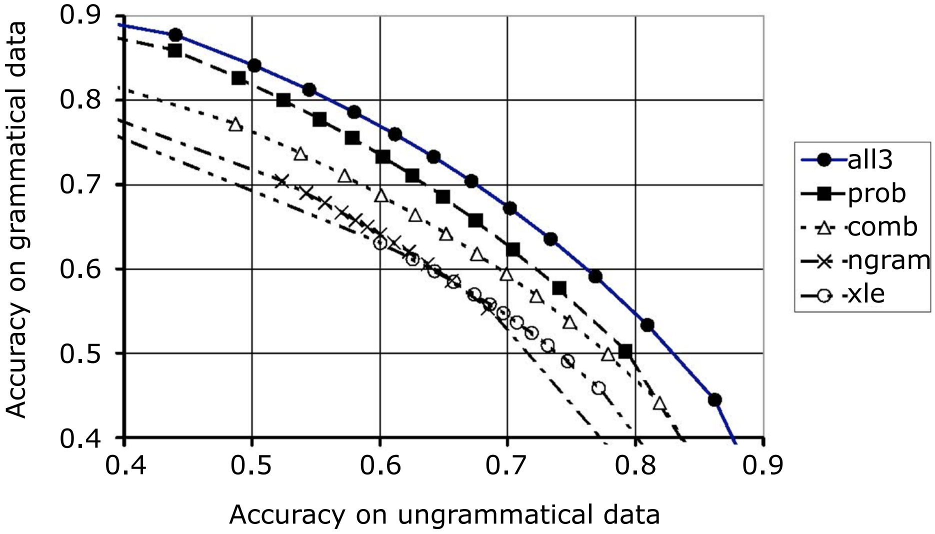
Evaluation Procedure

We measure the accuracy of our voting classifiers separately on grammatical and ungrammatical data. Therefore, each result is a point in a plane. A classifier is clearly better if it has higher accuracy on both scales, that is, if the point falls in the top right-hand corner relative to a point representing another classifier. Varying the voting threshold produces a number of points for each method that can be connected to a curve on which the accuracy trade-off can be chosen.

Artificial Test Data Results

Figure 1 shows how the different methods perform on BNC test data (ungrammatical side artificially created).

Figure 1
Accuracy Trade-Off with Voting Classifiers over 12 Decision Trees Measured on BNC Test Data (for different methods)



The **all3** method is the method that we obtain from combining the three feature sets of the **xle**, **ngram** and **prob** methods. Two possible combinations of the basic methods (**prob**/**xle** and **prob**/**ngram**) have been omitted to keep the graph readable. The connecting lines represent classifiers that could be built using interpolation[8] (including the two trivial classifiers that flag everything or nothing). The results shown are the average over 10 cross-validation runs.

Cross-validation and significance

Due to the high number of test sentences (4 million), there can be no doubt that the observed differences are statistically significant. To give an example, we calculate the p-value for the significance of the difference in accuracy on grammatical test data between methods **all3** and **prob** for the lowest voting threshold, see the two top left most data points in Figure 1. A randomized test (also called exact test) is computationally expensive for large test sets. Therefore, we find the p-value of p = .0003 for 10,000 test items with an approximate randomization test (100,000 iterations). It should be noted though that results between cross-validation runs vary along the accuracy trade-off curve. This is due to building decision trees that optimize for overall accuracy only.

Discussion

The combination of n-gram and xle features (**comb**) outperforms the individual **ngram** and **xle** methods as has previously been shown by Wagner et al. (2007). However, our new probabilistic method (**prob**) achieves even higher accuracy just on its own. It does particularly well on grammatical data. The combination of all three feature sets (**all3**) further improves results. The improvement over comb is similar on both ends of the accuracy trade-off curve.

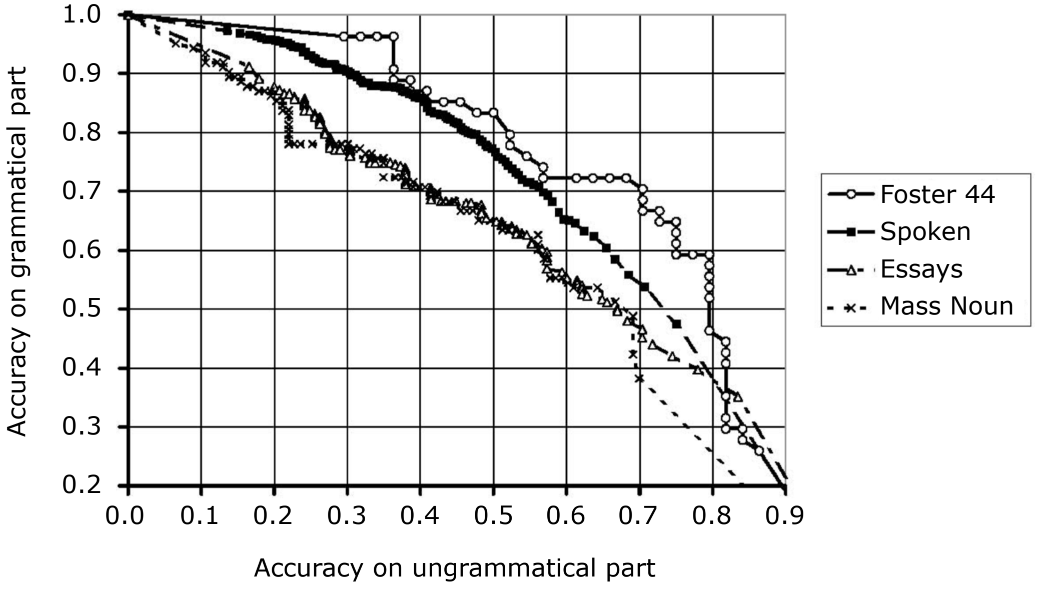
It is difficult to directly compare our results to those of Andersen (2007), Okanohara and Tsujii (2007), and Sun et al. (2007) because all four approaches use different test and training data: Andersen (2007) uses data from the Cambridge Learner Corpus, Okanohara and Tsujii (2007) use BNC sentences as grammatical data and synthetic ungrammatical data created by sampling BNC sentences with a trigram language model, and Sun et al. (2007) use text from the *Japan Times* and the "21st Century newspaper" as grammatical data and sentences from the Japanese Learners of English (JLE) corpus and the Chinese Learner Error Corpus (CLEC) as ungrammatical text. The accuracy of our method is within the same range as that of Andersen (2007) and Okanohara and Tsujii (2007) but falls short of the accuracy of 81.75% reported by Sun et al. (2007).

Learner Data Results

We test our combined method **all3** on the real learner data described above. We cannot report accuracy for sentences classified as "questionable" by our annotator, but it is reassuring that these sentences are flagged as ungrammatical more often than grammatical sentences and less often than ungrammatical sentences.

Figure 2 shows that we lose some accuracy when we switch from artificial test data to real data.

Figure 2
Accuracy Trade-Off with Voting Classifier **all3** over 120 Decision Trees Measured on Three Learner Corpora and the **Foster 44** test set—Number of sentences (ungrammatical/grammatical): **Essays**: 145/350, **Spoken**: 4285/500, **Mass Noun**: 123/123, **Foster 44**: 44/54



Method **all3** performs best on the held-out section of the corpus of naturally occurring errors that informed our automatic error insertion procedure (**Foster 44**). In contrast, the results for **Essays** and **Mass Noun** data are poor. At 70% accuracy on the grammatical side of the corpora, the baseline of randomly flagging 30% of all sentences is surpassed by only 10 percentage points to 40% accuracy on ungrammatical data. The results for **Spoken** learner data are much better. Here, 57% accuracy is reached under the same conditions. At 95% accuracy on grammatical data, over 20% of ungrammatical **Spoken** sentences are identified, more than 4 times over the 5% baseline.

Analysis

The drop in accuracy observed when moving from synthetic test data (Figure 1) to real test data (Figure 2) confirms the well known machine-learning dictum that training and test data should be as similar as possible. The best results for the real test data come from the **Foster 44** corpus which has a similar distribution of error types as the synthetic training data. The low results for the **Mass Noun** data can easily be explained by the absence of this type of error from our training data. Sun et al. (2007) also report a large drop in accuracy (from approximately 82% to 58%) when they apply a classifier trained on Chinese English data to Japanese English test data.

The difference between the **Essays** and **Spoken** test sets might be due to the source of the grammatical sentences that are used to plot the accuracy curve. The grammatical essay sentences are produced by learners themselves along with the ungrammatical sentences, while the transcribed spoken sentences are corrected by a native speaker. It is possible that the level of the learner also plays a role here. The sentences in the **Essays** test set have been produced by advanced learners, whereas the sentences in the **Spoken** test set have been produced by learners of various levels. Inspecting the breakdown by learner level in the **Spoken** test set confirms this: accuracy decreases as the learner level increases.

CONCLUSION

We have presented a new method for judging the grammaticality of a sentence which makes use of probabilistic parsing with treebank-induced grammars. Our new method exploits the differences between parse results for grammars trained on grammatical, ungrammatical, and mixed treebanks. The method combines well with n-gram and deep grammar methods in a machine-learning-based framework. In addition, voting classifiers have been proposed to tune the accuracy trade-off. This provides an alternative to the common practice of applying n-gram filters to increase the accuracy on grammatical data (Gamon et al., 2008; Lee & Seneff, 2008).

Our method was trained on sentences from the BNC and artificially distorted versions of these sentences produced using an error creation procedure. When tested on real learner data, we found that the method's accuracy drops, indicating that the next step in our research is to refine the error creation procedure to take into account a broader class of errors, including, for example, preposition errors and mass noun errors. In addition, we intend to experiment with adding noncontinuous sequential patterns as used by Sun et al. (2007) to our n-gram method to see if this improves performance. Another interesting future direction is to explore the relationship between our work and the machine-learning-based methods used in the machine translation community to evaluate the fluency of machine translation system output (Albrecht & Hwa, 2007). The area of research concerned with automatically evaluating writing style might also provide useful insights.

NOTES

[1] The tendency of the error creation procedure to produce covert errors was estimated by carrying out a small experiment involving 500 BNC sentences (Foster, 2007). According to this experiment, the error creation procedure will produce covert errors approximately 8% of the time, with the introduction of a missing word error most likely to result in a superficially well formed sentence (e.g., *She steered* ***Melissa*** *round a corner* → *She steered round a corner*).

[2] One exception is a noun phrase containing a list of coordinated adjectives, for example, *the green, white and [orange] tricolour*.

[3] We are very grateful to James Hunter from Gonzaga University for providing us with these data.

[4] These sentences are available for download (http://research.microsoft.com/research/downloads).

[5] The J48 learner of the weka machine learning package (Witten & Frank, 2000) with the minimum size of leaves set to 125 was used.

[6] This number was chosen due to time constraints. Training a tree on an Intel Core2 Duo processor takes between 30 minutes and 2 hours depending on the size of the feature set, and training has to be repeated for each set. Ideally, a large number of trees should be employed.

[7] Ten trees are trained on a sliding window of 5/18 = 27.8% of the training data available to a cross-validation run (i.e., 25% of the overall data). Each window overlaps with its neighbors by (5 - (18 - 5)/9)/5 = 71.1%.

[8] An interpolating classifier randomly chooses for each item to be classified among a set of classifiers and outputs its prediction. In the case of two classifiers *A* and *B*, there is only one parameter *p* by which *A* is chosen. (*B* is chosen with probability 1 - *p*.) Let a_1, …, a_n and b_1, …, b_n be the accuracies of *A* and *B* on *n* test sets (*n* = 2 in our evaluation). Then the expectation values of the accuracies of the interpolating classifier are $c_i = p$ X a_i + (1- p) X b_i for i = 1, …, n, that is, they fall on the line connecting a_i and b_i.

REFERENCES

Albrecht, J. S., & Hwa, R. (2007). A re-examination of machine learning approaches for sentence-level MT evaluation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics* (pp. 880-887). Prague, Czech Republic: Association for Computational Linguistics.

Andersen, O. E. (2007). Grammatical error detection using corpora and supervised learning. In V. Nurmi & D. Sustretov (Eds.), *Proceedings of the Twelfth ESSLLI Sudent Session* (pp. 1-9). Dublin, Ireland.

Bauer, E., & Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning, 36*(1-2), 105-139.

Bender, E. M., Flickinger, D., Oepen, S., & Baldwin, T. (2004). Arboretum: Using a precision grammar for grammar checking in CALL. In R. Delmonte, P. Delcloque, & S. Tonelli (Eds.), *Proceedings of the InSTIL/ICALL Symposium: NLP and Speech Technologies in Advanced Language Learning Systems* (pp. 83-86). Venice, Italy.

Bigert, J. (2004). Probabilistic detection of context-sensitive spelling errors. In *Proceedings of the 4th International Conference on Language Resources and Evaluation* (Vol. 5, pp. 1633-1636). Lisbon, Portugal: European Language Resources Association.

Bigert, J., & Knutsson, O. (2002). Robust error detection: A hybrid approach combining unsupervised error detection and linguistic knowledge. In *Proceedings of the 2nd Workshop on Robust Methods in Analysis of Natural Language Data (Romand)*. Frascati, Italy. Retrieved May 4, 2009, from http://www.johnnybigert.se/publications.html

Bigert, J., Sjöbergh, J., Knutsson, O., & Sahlgren, M. (2005). Unsupervised evaluation of parser robustness. In A. Gelbukh (Ed.), *Proceedings of the Sixth International Conference on Intelligent Text Processing and Computational Linguistics (CICling)* (pp. 142-154). Mexico City, Mexico: Springer.

Black, E., Abney, S., Flickenger, S., Gdaniec, C., Grishman, C., Harrison, P., et al. (1991). Procedure for quantitatively comparing the syntactic coverage of English grammars. In E. Black (Ed.), *Proceedings of the HLT Workshop on Speech and Natural Language* (pp. 306-311). Morristown, NJ: Association for Computational Linguistics.

Breiman, L. (1996). Heuristics of instability and stabilization in model selection. *Annals of Statistics, 24*(6), 2350-2383.

Brockett, C., Dolan, W. B., & Gamon, M. (2006). Correcting ESL errors using phrasal SMT techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics* (Vol. 1, pp. 249-256). Sydney, Australia: Association for Computational Linguistics.

Burnard, L. (2000). *User reference guide for the British National Corpus* (Technical Report). Oxford University Computing Services.

Butt, M., Dyvik, H., King, T. H., Masuichi, H., & Rohrer, C. (2002). The parallel grammar project. In *Proceedings of COLING-2002 workshop on Grammar Engineering and Evaluation* (pp. 1-7). Morristown, NJ: Association for Computational Linguistics.

Charniak, E., & Johnson, M. (2005). Course-to-fine n-best-parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics* (pp. 173-180). Ann Arbor, Michigan: Association for Computational Linguistics.

De Felice, R., & Pulman, S. (2008). A classifier-based approach to preposition and determiner error correction in L2 English. In *Proceedings of COLING* (pp. 169-176). Manchester, UK: Coling 2008 Organizing Committee.

Foster, J. (2005). *Good reasons for noting bad grammar: Empirical investigations into the parsing of ungrammatical written English*. Unpublished doctoral dissertation, Trinity College, University of Dublin.

Foster, J. (2007). Treebanks gone bad: Parser evaluation and retraining using a treebank of ungrammatical sentences. *International Journal on Document Analysis and Recognition, 10*(3-4), 129-145.

Foster, J., Wagner, J., & Genabith, J. van. (2008). Adapting a WSJ-trained parser to grammatically noisy text. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: ACL-HLT-08 short papers volume* (pp. 221-225). Columbus, OH: Association for Computational Linguistics.

Gamon, M., Gao, J., Brockett, C., Klementiev, A., Dolan, W. B., Belenko, D., et al. (2008). Using contextual speller techniques and language modeling for ESL error correction. In *Proceedings of the Third International Joint Conference on Natural Language Processing* (pp. 449-455). Hyderabad, India: Asian Federation of Natural Language Processing.

Golding, A. R., & Schabes, Y. (1996). Combining trigram-based and feature-based methods for context-sensitive spelling correction. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics* (pp. 71-78). Santa Cruz, CA: Association for Computational Linguistics.

Granger, S. (1993). International corpus of learner English. In J. Aarts, P. de Haan, & N. Oostdijk (Eds.), *English language corpora: Design, analysis and exploitation* (pp. 57-71). Amsterdam: Rodopi.

Han, N.-R., Chodorow, M., & Leacock, C. (2006). Detecting errors in English article usage by non-native speakers. *Natural Language Engineering, 12*(2), 115-129.

Horváth, J. (1999). *Advanced writing in English as a foreign language: A corpus-based study of processes and products*. Unpublished doctoral dissertation, Janus Pannonius University, Pécs, Hungary.

James, C. (1998). *Errors in language learning and use: Exploring error analysis*. London: Addison Wesley Longman.

Kukich, K. (1992). Techniques for automatically correcting words in text. *ACM Computing Surveys, 24*(4), 377-439.

Lee, J., & Seneff, S. (2008). Correcting misuse of verb forms. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics* (pp. 174-182). Columbus, OH: Association for Computational Linguistics.

Marcus, M., Kim, G., Marcinkiewicz, M. A., MacIntyre, R., Bies, A., Ferguson, M., et al. (1994). The Penn treebank: Annotating predicate argument structure. In C. J. Weinstein (Ed.), *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey* [the 1994 ARPA Human Language Technology Workshop] (pp. 114-119). Princeton, NJ: Morgan Kaufmann.

Maxwell, J., & Kaplan, R. (1996). Unification-based parsers that automatically take advantage of context freeness. In M. Butt & T. H. King (Eds.), *Proceedings of the First International Conference on Lexical Functional Grammar*. Grenoble, France. Retrieved May 4, 2009, from http://www.parc.com/research/publications/details.php?id=3115

Okanohara, D., & Tsujii, J. (2007). A discriminative language model with pseudo-negative samples. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics* (pp. 73-80). Prague, Czech Republic: Association for Computational Linguistics.

Pelcra: Polish and English language corpora for research and applications. (2004). Retrieved November 9, 2004, from http://pelcra.ia.uni.lodz.pl

Sampson, G., & Babarczy, A. (2003). A test of the leaf-ancestor metric for parse accuracy. *Natural Language Engineering, 9*(4), 365-380.

Sjöbergh, J. (2006). Chunking: An unsupervised method to find errors in text. In S. Werner (Ed.), *Proceedings of the 15th Nodalida Conference, Joensuu 2005* (pp. 180-185). Joensuu, Finland: University of Joensuu electronic publications in linguistics and language technology.

Smith, N. A., & Eisner, J. (2005a). Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting of the Association of Computational Linguistics* (pp. 354-362). Ann Arbor, Michigan: Association for Computational Linguistics.

Smith, N. A., & Eisner, J. (2005b). Guiding unsupervised grammar induction using contrastive estimation. In C. de la Higuera, T. Oates, G. Paliouras, & M. van Zaanen (Eds.), *Proceedings of the IJCAI workshop on Grammatical Inference Applications* (pp. 73-82). Edinburgh, Scotland.

Snow, C., & Meijer, G. (1976). On the secondary nature of syntactic intuitions. In S. Greenbaum (Ed.), *Acceptability in language* (pp. 163-177). The Hague: Mouton.

Sun, G., Liu, X., Cong, G., Zhou, M., Xiong, Z., Lee, J., et al. (2007). Detecting erroneous sentences using automatically mined sequential patterns. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics* (pp. 81-88). Prague, Czech Republic: Association for Computational Linguistics.

Tetreault, J., & Chodorow, M. (2008a). Native judgments of non-native usage: Experiments in preposition error detection. In *Proceedings of the Coling 2008 Workshop on Human Judgments in Computational Linguistics* (pp. 24-32). Manchester, UK: Coling 2008 Organizing Committee.

Tetreault, J. R., & Chodorow, M. (2008b). The ups and downs of prepositions. In *Proceedings of COLING* (pp. 865-872). Manchester, UK: Association for Computational Linguistics.

Verberne, S. (2002). *Context-sensitive spell checking based on word trigram probabilities*. Unpublished master's thesis, University of Nijmegen.

Wagner, J., Foster, J., & Genabith, J. van. (2007). A comparative evaluation of deep and shallow approaches to the automatic detection of common grammatical errors. In *Proceedings of the Joint International Conference on Empirical Methods in Natural Language Processing (EMNLP) and Natural Language Learning (CoNLL)* (pp. 112-121). Prague, Czech Republic: Association for Computational Linguistics.

Witten, I. H., & Frank, E. (2000). *Data mining: Practical machine learning tools and techniques with java implementations*. San Mateo, CA: Morgan Kaufmann.

ACKNOWLEDGMENTS

This work was supported by the IRCSET Embark Initiative (postdoctoral fellowship P/04/232). The training data and the artificial test data used in this research is based on the British National Corpus (BNC), distributed by Oxford University Computing Services on behalf of the BNC Consortium. We thank Djamé Seddah for helping us to parse the corpora on the SFI/HEA Irish Centre for High-End Computing (ICHEC), and we wish to acknowledge ICHEC for the provision of computational facilities and support. We are very grateful to James Hunter of Gonzaga University for providing us with the spoken error corpus. Finally, we would like to thank the reviewers for their very helpful comments.

AUTHORS' BIODATA

Joachim Wagner is full-time System Administrator for the Centre for Next Generation Localisation (CNGL) at Dublin City University. He studied Computational Linguistics and Artificial Intelligence at the Universität Osnabrück (Germany) where he graduated in 2003. During the last 3 years in Osnabrück, he worked part time as Assistant Developer for the LogoTax project which included the implementation of a web-based lexicographic corpus tool for computer-assisted language learning. Wagner was awarded an IRCSET scholarship in 2003 to pursue a Ph.D. in the area of computer-assisted language learning at Dublin City University. He is currently writing up his Ph.D. research on detecting grammatical errors with probabilistic parsing.

Jennifer Foster is the Director of the National Centre for Language Technology at the School of Computing in Dublin City University (http://nclt.computing.dcu.ie), where she has worked as a researcher since 2006. She obtained a B.A. in Computer Science, Linguistics and German and a Ph.D. in Computer Science from Trinity College, Dublin. She spent a year studying in Bielefeld University (Germany) and has worked for two technology startup companies. Her research interests include robust parsing, parser evaluation, grammatical error detection, natural language generation, and sentiment analysis.

Josef van Genabith is the Director of the Centre for Next Generation Localisation (CNGL, http://www.cngl.ie). He obtained his first degree from RWTH Aachen (Germany), a Ph.D. from the University of Essex (UK), worked as a researcher at the University of Stuttgart (Germany), and is Professor in the School of Computing in Dublin City University (Ireland). His research is on multilingual treebank-based wide-coverage LFG grammar acquisition, parsing and generation with those grammars, machine translation, semantics, CALL, and localization.

AUTHORS' ADDRESS

Dublin City University
School of Computing
Glasnevin
Dublin 9
Ireland
Phone: +353 1 700 6915 (Joachim Wagner)
+353 1 700 5263 (Jennifer Foster)
+353 1 700 5074 (Josef van Genabith)
Email: jwagner@computing.dcu.ie
jfoster@computing.dcu.ie
josef@computing.dcu.ie