Robust Error Detection: A Hybrid Approach Combining Unsupervised Error Detection and Linguistic Knowledge

Johnny Bigert and **Ola Knutsson**
Numerical Analysis and Computer Science
Royal Institute of Technology, Sweden
{johnny, knutsson}@nada.kth.se

Abstract

This article presents a robust probabilistic method for the detection of context-sensitive spelling errors. The algorithm identifies less-frequent grammatical constructions and attempts to transform them into more-frequent constructions while retaining similar syntactic structure. If the transformations result in low-frequency constructions, the text is likely to contain an error. A first unsupervised approach uses only information derived from a part-of-speech tagged corpus. This experiment shows a good error detection capacity but also a high rate of false alarms, in many cases due to phrase and clause boundaries. In a second approach, we combine the first method with robust phrase and clause recognition to avoid many of the false alarms in the first experiment. A comparative evaluation of the experiments shows that the introduction of linguistic knowledge dramatically increases the precision of the error detection method.

1 Introduction

Even though spell checkers are widely used and commercialized, many spelling mistakes in writer's texts are still left for humans to identify. In this paper we focus on a category of error types that we consider very difficult to detect. One of these error types is the category of so-called context-sensitive spelling errors (e.g. Mays et al. (1991)). The other error type that we are focusing on is erroneously split compounds which are frequent in compounding languages.

Several approaches have been proposed to detect and correct context-sensitive spelling errors. Most approaches operate on sets of easily confused words and are based on context features for each word in the confusion set, such as word and parts-of-speech context (Yarowsky (1994), Golding (1995), Golding and Roth (1996)). Furthermore, some include part-of-speech (POS) tag trigram information to determine which candidate is the most likely (Mays et al. (1991), Golding and Schabes (1996)). A rule-based approach using machine learning is given in (Mangu and Brill, 1997). Although useful for detection and correction, these approaches require a list of confusion sets predicted beforehand.

The main drawback with the algorithms mentioned for error detection is that they require knowledge about the errors to be found. Often, such errors are not known in advance and the errors predicted may not be sufficient. We want to be able to detect errors from categories of difficult spelling errors such as spelling errors resulting in an existing word. Furthermore, we would like something more general and robust.

In this paper we propose a probabilistic method for detection of context-sensitive spelling errors and erroneously split compounds. The main contribution of this paper is an approach to mitigate the problem of sparse data. To this end, we use a novel combination of existing techniques, such as POS tagging, shallow parsing and phrase transformations.

The basic idea is to identify rare sequences of morpho-syntactic tags and by different methods determine if the sequences are rare due to the sparse data problem or phrase- and clause boundaries. If a rare sequence cannot be transformed into a more frequent one using the methods, the sequence is considered to contain an error. We have investigated the methods in two experiments.

The first experiment is conducted with an unsupervised method using only information derived from a rather small part-of-speech tagged corpus of 1 million words (Ejerhed et al., 1992). To deal with the sparse data problem we use a distance metric for part-of-speech tags (Bigert, 2002) in order to replace rare tags with tags used in similar syntactic contexts.

The second experiment is using the method described above, complemented with a robust phrase and clause recognizer (Knutsson et al., forthcoming) in order to deal with phrase and clause boundaries that often contain rare sequences of morpho-syntactic tags. Using the information of clause boundaries, clauses are used as the unit for the error detection algorithm to operate on. The phrase recognition is used to transform rare phrase boundaries into more frequent and by doing so avoiding false alarms. The experiments have been conducted on authentic texts written in Swedish.

In this paper, we focus on the detection of errors and do not address the intricate problem of error classification or correction.

1.1 Algorithm Outline

Part-of-speech tag n-grams have many useful properties. As the n-grams are extracted from a corpus representing the language, they capture some of the language's features. Because of the limited scope of an n-gram, the extracted features will contain only local information. Each of these n-grams constitutes a small acceptance grammar since it describes an acceptable sequence of n POS tags in the language. Altogether, the n-grams form a grammar containing local information about the acceptable grammatical constructs of the language. In contrast, POS tag n-grams not in the grammar are considered ungrammatical. From these observations, we will construct a simple error detection algorithm.

This naive algorithm has a few drawbacks, all caused by the same problem: insufficient POS tag data. The lack of data entails that many grammatical constructs may never have been encountered. In the POS tag case, we alleviate this effect by using the n-gram frequency table once again. From the frequencies we build a matrix of syntactic distances between POS tags called the confusion matrix of representatives.

The confusion matrix contains information on how suitable one tag is in the context of another. The matrix provides us with a means to replace rare tags with more frequent tags. From this, we construct an algorithm improving the naive. This method is covered in Section 2.

The improved algorithm will still suffer from the problem of sparse data, mainly due to rare n-grams resulting from clause and phrase boundaries. We approach this problem by transforming rare phrase constructs to those more frequent. The aim is that the transformation will produce a high-frequency n-gram if the original construct was grammatical. This is covered in Section 3.

2 Error Detection with no Linguistic Knowledge

In this section, we present a probabilistic method for detection of spelling errors, not requiring any previous knowledge of error types. The algorithm is based upon statistics from a corpus of correct and balanced text.

2.1 Detection of Improbable Grammatical Constructs

The POS tag n-gram table includes the frequency of each n-gram. As indicated in the algorithm outline, the n-grams constitute a local grammar, and from that, we can devise a rudimentary error detection algorithm. An implementation for trigrams is shown in Algorithm 1.

```
Algorithm 1: ProbCheck
Description: A first approach to a probabilistic error detector
Input: A tag stream $\bar{s}_k = (t_1, t_2, \ldots, t_k)$ and a grammaticality threshold $e$.
Output: A set of indexes of ungrammatical constructs if found, $\emptyset$ (the empty set) otherwise.
ProbCheck($\bar{s}_k, e$)
(1)  $I \leftarrow \emptyset$
(2)  foreach $i$ in $[2, k-1]$
(3)      if TriFreq($t_{i-1}, t_i, t_{i+1}$) $< e$
(4)          $I \leftarrow I \cup \{i\}$
(5)  return $I$
```

The text to be scrutinized must first be tagged with a POS tag disambiguator. From the resulting tag stream, each n-gram is looked up in the n-gram frequency table. If the frequency exceeds a pre-determined threshold, the construct is considered grammatically sound. Otherwise, it is a rare or incorrect grammatical construct, and therefore improbable to be the intention of the writer. Thus, the n-gram is flagged as a potential grammatical error.

One serious problem concerning this approach is rare constructs due to insufficient data and infrequent tags. An n-gram representing an acceptable grammatical construct may not have been encountered because of the rareness of the tags participating in the n-gram. This effect can be mitigated by using the confusion matrix introduced next.

2.2 Sparse Data and the Confusion Matrix

The idea behind the *confusion matrix of representatives* is to find suitable representatives for rare tags since rare tags often result in low-frequency n-grams. We use an example to illustrate the problem with rare grammatical constructs from the previous section.

Say that we have encountered a sentence in Swedish "Det är varje chefs uppgift att..." (It is every manager's responsibility to...). The tag disambiguator has tagged the part "det är varje" (it is every) with (*pn.neu.sin.def.sub/obj, vb.prs.akt, dt.utr/neu.sin.ind*). A look-up in the trigram frequency table reveals that this particular trigram has never been encountered before even though the construction is grammatically sound. This may be attributed to the fact that one of the participating tags have low frequency and in this example, the third tag (*dt.utr/neu.sin.ind*) is rare with only 704 occurrences (0.07% out of a million words). A language construct, very much similar in meaning to the one above, is "det är en" (it is a) with tags (*pn.neu.sin.def.sub/obj, vb.prs.akt, dt.utr.sin.ind*). This small change in meaning increases the individual tag frequency from 704 occurrences for (*dt.utr/neu.sin.ind*) to 19112 occurrences for (*dt.utr.sin.ind*). The trigram frequency rises from 0 occurrences for (*pn.neu.sin.def.sub/obj, vb.prs.akt, dt.utr/neu.sin.ind*) to 231 occurrences for (*pn.neu.sin.def.sub/obj, vb.prs.akt, dt.utr.sin.ind*). We see that replacing (*dt.utr/neu.sin.ind*) with (*dt.utr.sin.ind*) reduces the problem with rare tags while retaining almost the same meaning. Explanation of the tags used is given in Table 1.

The example indicates that we could benefit from substituting a rare tag with a tag of higher frequency suitable in the same context. If substituting tag t_1 with tag t_2, we shall call t_2 a *representative* for t_1.

Even though a person can easily produce a list of feasible representatives for each tag, one problem is the order between the representatives. Also, not all representatives are equally appropriate given all contexts and thus, a weight will be needed.

One approach to produce such a list is to use the L1-norm and POS tag n-grams to measure the distance between two tags (Bigert, 2002). If we use trigrams as an example, the distance is calculated as explained below.

We are given a tag t, a replacement tag t' and two context tags t_L and t_R. First, we normalize the trigram (t_L, t, t_R) to obtain a fair comparison between tags of different frequency:

$$n(t_L, t, t_R) = \frac{freq(t_L, t, t_R)}{freq(t)}.$$

Second, we calculate the difference between the normalized frequencies:

$$dist_{t_L,t_R}(t, t') = \left| n(t_L, t, t_R) - n(t_L, t', t_R) \right|.$$

Last, we consider all POS tag contexts:

$$dist(t, t') = \sum_{t_L, t_R} dist_{t_L,t_R}(t, t').$$

We get that $dist(t, t')$ ranges from 0 (where the contexts are identical) to 2 (where the uses of t and t' are disjunct). From this, we calculate the probability $p(t, t')$ of replacing t with t', where a probability of 0 is totally disjunct syntactic uses and 1 means essentially the same tag. We define the *confusion matrix of representatives* Δ from these probabilities by assigning $\Delta = \{p(t_i, t_j)\}$ for all tags t_i, t_j. We denote the m best representatives for a tag t by $r(t, 1), r(t, 2), \ldots, r(t, m)$, where $r(t, 1)$ is the best candidate.

2.3 Weighted n-grams

Given the confusion matrix, we now have the tools to replace rare tags with their representatives. When a tag n-gram of low frequency is encountered, we want to determine whether the low frequency is due to ungrammaticality or merely the low frequency of the participating tags. Hence, we want to determine whether substituting one of the tags may increase the frequency. When substituting one tag with its representative, we must take into consideration the syntactic distance between the tags involved, when calculating the new n-gram frequency resulting from the switch.

For example, given the trigram (t_1, t_2, t_3) with frequency $f = freq(t_1, t_2, t_3)$, we use the tag r_1 to replace t_1. From the confusion matrix we get a probability of $q = p(t_1, r_1)$. A q of 1 would imply that t_1 and r_1 are used in identical syntactic contexts and thus, no penalty should be imposed. A $q < 1$ implies that the use of t_1 and r_1 differs, and a penalty is in order since there is a probability that the use of r_1 in this context may be less appropriate than the use of t_1. We calculate the new trigram frequency for (r_1, t_2, t_3) as $f' = freq(r_1, t_2, t_3) \cdot q$, that is, the new trigram frequency penalized. If f' is above a given frequency threshold, thus improving on the old trigram frequency, the construct is considered grammatically sound.

When substituting more than one tag simultaneously we take into consideration all syntactic distances involved by defining the compound penalty $w(\bar{t}, \bar{r}) = \prod_{i=1}^{n} p(t_i, r_i)$, where $\bar{t} = (t_1, t_2, \ldots, t_n)$, $\bar{r} = (r_1, r_2, \ldots, r_n)$ and r_j is a representative for t_j.

We now construct a measure of the probability of grammaticality when given an n-gram tag sequence. The intention here is to consider all m representatives for each of the n tags in the n-gram. (Note that the tag itself is included among the representatives.) Each of these m^n new n-grams has a frequency which indicates their individual grammaticality. This frequency is weighted using the compound penalty above to compensate for the syntactic distance between the tags and their representatives.

Definition: *The* weighted n-gram frequency *of an n-gram sequence $\bar{t} = (t_1, t_2, \ldots, t_n)$ is defined as*

$$wfreq(\bar{t}) = \sum_{\bar{r} \in R} w(\bar{t}, \bar{r}) \cdot freq(\bar{r}),$$

where $R = \{(r(1, i_1), r(2, i_2), \ldots, r(n, i_n)\}_{\bar{i} \in [m]^n}$ is the set of all m^n representatives for $\bar{t}$.

The intuition behind the weighted frequency is simply to attempt all different combinations of replacements for the tags in the n-gram. We will use the weighted frequency as a measurement of the grammaticality of a sentence. If the weighted frequency of any of the n-grams is below a given threshold, that part is considered ungrammatical.

2.4 The algorithm

The final algorithm is implemented for trigrams in Algorithms 2 and 3. Algorithm 2 is very similar to Algorithm 1 but utilizes weighted trigrams. In Algorithm 3, the compound penalty is computed over R containing $m^n = m^3$ representatives. For each representative, the penalty w is computed on lines 5–6 and the trigram frequency at lines 7–8.

Algorithm 2: ProbCheck
Description: The improved probabilistic error detector
Input: A tag stream $\bar{s}_k = (t_1, t_2, \ldots, t_k)$; a grammaticality threshold e; a confusion matrix of representatives Δ; representatives $R_m = \{r(t, i)\}$, denoting the ith representative of t where t is a POS tag and where $i = 1, 2, \ldots, m$.
Output: A set of indexes of ungrammatical constructs if found, $\emptyset$ (the empty set) otherwise.

```
ProbCheck($\bar{s}_k, e, \Delta, R_m$)
(1)  $I \leftarrow \emptyset$
(2)  foreach $i$ in $[2, k-1]$
(3)      if WtTriFreq($t_{i-1}, t_i, t_{i+1}$,
(4)                       $\Delta, R_m$) $< e$
(5)          $I \leftarrow I \cup \{i\}$
(6)  return $I$
```

3 Error Detection Using Linguistic Knowledge

The main problem with the probabilistic error detection is the fact that phrase and clause boundaries may involve almost any POS tag n-gram and thus, many n-grams may never have been encountered. In this section, we make use of phrases and clause boundaries to remove false alarms resulting from such boundaries.

3.1 Clause and Phrase Recognition

The identification of clause and phrase boundaries is important for syntactic analysis. For example, the recognition of clause boundaries is an essential and repeated step in Constraint

Algorithm 3: WtTriFreq
Description: Calculate weighted trigram frequencies
Input: A tag trigram t_1, t_2, t_3. For Δ and R_m, see Algorithm 2.
Output: The weighted trigram frequency of the trigram provided

```
WtTriFreq($t_1, t_2, t_3, \Delta, R_m$)
(1)  $sum \leftarrow 0$
(2)  foreach $i_1$ in $[1, m]$
(3)      foreach $i_2$ in $[1, m]$
(4)          foreach $i_3$ in $[1, m]$
(5)              $penalty \leftarrow$
(6)                  $\prod_{j=1}^{3} \Delta(t_j, r(t_j, i_j))$
(7)              $freq \leftarrow$ TriFreq
(8)                  $(r(t_1, i_1), r(t_2, i_2), r(t_3, i_3))$
(9)              $sum \leftarrow$
(10)                 $sum + penalty \cdot freq$
(11) return $sum$
```

Grammar parsing (Karlsson et al., 1995). In the system described here the recognition of clauses and phrases plays an important role. The error detection method proposed is independent of how the recognition is accomplished. We have chosen to implement a rule-based phrase and clause identifier, even though an unsupervised and probabilistic method would suffice. The most important property is robustness.

For the detection of clause boundaries, we have implemented Ejerhed's algorithm for Swedish (Ejerhed, 1999). This algorithm is based on context-sensitive rules operating on POS tags. One main issue is to disambiguate conjunctions that can coordinate words in phrases, whole phrases and, most important, clauses. About 20 rules were implemented for the detection of clause boundaries in the Granska framework (Domeij et al., 2000).

The module for phrase recognition should identify the "best" phrase candidates and assign them with the head's feature values. For example, a noun phrase of the type "den lilla pojken" (the little boy) is assigned with the following features and values: word class is noun, gender is non-neuter, number is singular, species is definite, case is nominative. This results in a valid tag (*nn.utr.sin.def.nom*), corresponding to the head "pojken" (the boy). The assignment must result in one or more valid tags to be useful to the probabilistic error detection algorithm. Furthermore, some constructs may be removed from the analyzed text (e.g. prepositional and adverbial phrases), which is motivated by the observation that removal of such phrases very seldom violates the syntax of the language. The rules implemented are liberal regarding the syntactic agreement within the phrase. We have chosen this strategy for several reasons. First, we want to analyze sentences that may contain one or more errors. Second, the linguistic rules for agreement in Swedish contain some problematic exceptions. Third, tagging errors from the part-of-speech tagger could cause insufficient disambiguation of phrase boundaries.

3.2 Phrase Transformations for Rare n-grams

We aim to produce a sentence representation without rare n-grams while retaining grammaticality and preferably meaning similar to the original sentence. As explained in the previous section, every phrase may be replaced by zero or more tags. Zero tags will remove the phrase (e.g. adverbial or prepositional phrases) and one or more tags will replace the phrase with the head (e.g. noun and verb phrases). The replacement of a phrase results in a longer scope for the probabilistic error detector. Furthermore, it replaces long and rare phrase constructs with the more common, minimal phrase consisting of the head only.

Algorithm 4: PhraseProbCheck
Description: The phrase enhanced probabilistic error detector
Input: A tag stream $\bar{s} = (t_1, t_2, \ldots, t_k)$ and a grammaticality threshold e.
Output: A set of indexes of ungrammatical constructs if found, $\emptyset$ (the empty set) otherwise.

```
PhraseProbCheck($\bar{s}, e$)
(1)  $I \leftarrow \emptyset$
(2)  foreach $i$ in $[2, k-1]$
(3)      if ProbCheck($i, \bar{s}$) $< e$
(4)          if not ClauseBound($i, \bar{s}$)
(5)              if not nGramOk($i, \bar{s}, e$)
(6)                  $I \leftarrow I \cup \{i\}$
(7)  return $I$
```

An implementation of the phrase enhanced probabilistic error detection for n-grams is given in Algorithm 4. At line 3, the tag stream is checked for grammatical errors. If no errors are found, the sentence is considered grammatical and the algorithm terminates. If a suspicious n-gram is found, part of the sentence will be subjected to further tests (line 5). The clause boundary condition is checked at line 4 so that detections adjacent to a clause boundary are not reported as errors. In Algorithm 5, we seek to resolve the problem with the rare n-gram found. At line 1 we identify the phrases overlapping the n-gram at index i. From these, we construct all combinations of phrases so that no two phrases span a common POS tag index (line 2). Given one of the combinations, we attempt to replace the participating phrases with their heads (line 4). If the n-gram at index i in the new tag stream is approved by the probabilistic error detection (line 5), we consider the n-gram grammatically sound. If none of the combinations result in an acceptable POS tag n-gram, a grammatical error is reported at line 7.

Algorithm 5: nGramOk
Description: The algorithm for phrase replacement and removal
Input: An index i containing a rare n-gram, a tag stream $\bar{s} = (t_1, t_2, \ldots, t_k)$ and a grammaticality threshold e
Output: True if the n-gram is grammatical, False otherwise

```
nGramOk($i, \bar{s}, e$)
(1)  $P \leftarrow$ OverlapPhrases($i, \bar{s}$)
(2)  $\mathcal{C} \leftarrow$ CombinePhrases($P$)
(3)  foreach $C$ in $\mathcal{C}$
(4)      $\bar{s}' \leftarrow$ Replace($C, \bar{s}$)
(5)      if ProbCheck($i, \bar{s}'$) $\geq e$
(6)          return True
(7)  return False
```

The use of the algorithm is best illustrated with an example (see Table 1 for an explanation of the tag set). Say that we have encountered the sentence "den lilla vasen på hyllan är inte så ful" (the little vase on the shelf is not so ugly) where the part "hyllan är inte" (shelf is not) is tagged (*nn.utr.sin.def.nom, vb.prs.akt.kop, ab*). The initial probabilistic test erroneously indicates an error and we construct the phrases overlapping the trigram centered at index 6 (see Figure 1):

1. Np: den lilla vasen på hyllan (the little vase on the shelf) $\rightarrow$ vasen (*nn.utr.sin.def.nom*) (the vase)
2. Pp: på hyllan (on the shelf) $\rightarrow$ *remove*
3. AdvP: inte så (not so) $\rightarrow$ *remove*

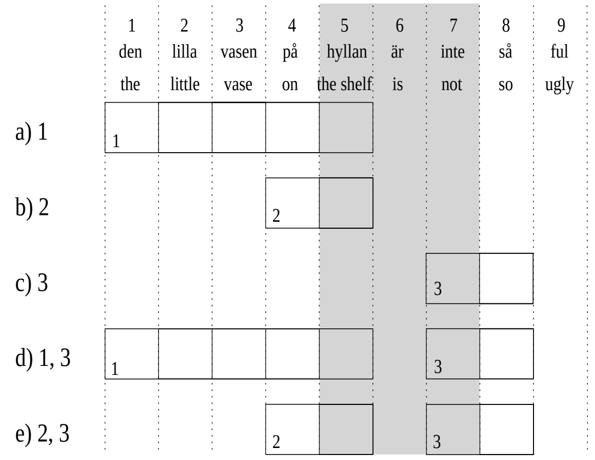

Figure 1: *Combination of phrases overlapping the suspicious trigram (highlighted).*

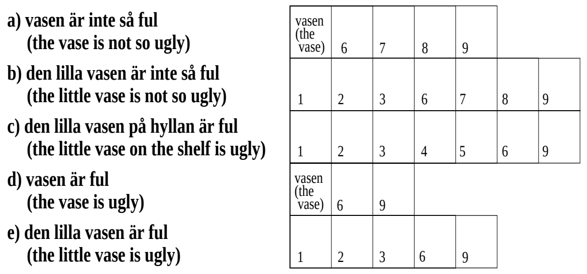
a) vasen är inte så ful (the vase is not so ugly)
vasen (the vase) 6 7 8 9
b) den lilla vasen är inte så ful (the little vase is not so ugly)
1 2 3 6 7 8 9
c) den lilla vasen på hyllan är ful (the little vase on the shelf is ugly)
1 2 3 4 5 6 9
d) vasen är ful (the vase is ugly)
vasen (the vase) 6 9
e) den lilla vasen är ful (the little vase is ugly)
1 2 3 6 9

Figure 2: *The resulting sentences from the combinations in Figure 1.*

From the phrases, we construct all combinations as shown in Figure 1, where the combination (1, 2) is not included due to the overlap between the two phrases. The resulting sentences are shown in Figure 2. Combinations **a** and **b** both produce rare trigrams due to the adverbial construction "inte så" (not so). Combination **c** removes the adverbial construction and produces an acceptable trigram. Throughout the replacements, grammaticality is retained, even though the content of the sentence may be somewhat altered, as seen in Figure 2.

Table 1: *Explanation of the tag set used. The tag set comprises 149 tags.*

noun (*nn*)	pronoun (*pn*)	verb (*vb*)	determiner (*dt*)	adverb (*ab*)
non–neuter (*utr*)	neuter (*neu*)	singular (*sin*)	plural (*plu*)	
definite (*def*)	indefinite (*ind*)	nominative (*nom*)	genitive (*gen*)	
present (*prs*)	active (*akt*)	copula (*kop*)		
subject (*sub*)	object (*obj*)			

4 Experiments and Evaluation

The experiments reported here have been conducted by using the Stockholm-Umeå corpus (SUC) (Ejerhed et al., 1992), consisting of one million words in Swedish. The corpus was used for extraction of n-grams and training of the POS tagger (below). The evaluation set consisted of 20000 words of school essays written by students, 16 to 18 years old.

The text was fully disambiguated using a hidden Markov model POS tagger for Swedish (Carlberger and Kann, 1999) with a performance of about 96.4% on unrestricted text. The tag set was a slightly modified version of the tag set from the training corpus (SUC) and comprised 149 tags. In a future setting, tag sets of different size and detail will be considered. Clearly, the tag set will affect both the tagger and the behavior of the error detection. Preliminary results on the performance of the shallow parser used for NP recognition is about 83.1% precision and 79.5% recall (Johansson, 2000). Performance when parsing other phrase types is yet to be evaluated. Preliminary results on the performance of the clause recognizer is 81.4% precision and 86.6% recall (not counting clause boundaries at sentence boundaries) (Hahne, forthcoming).

In the experiment setting, an interesting aspect is the number of representatives m to consider for each tag. We conducted experiments for several values but report only on the most promising, $m = 3$. Another aspect is the threshold for ungrammaticality for the weighted trigrams. This threshold can be set arbitrarily. Higher threshold will yield higher recall.

We wanted to assess how linguistic knowledge influences the performance of probabilistic error detection. To this end, the unsupervised algorithm was compared to one using linguistic knowledge in the form of clause and phrase structure identification. We ran three experiments for each algorithm to measure how the detections and false alarms were affected by linguistic knowledge.

4.1 Results

The distribution of errors resulting from the six experiments is shown in Table 2. The error types found by the algorithm are for example spelling errors and spelling errors resulting in an existing word, verb tense errors (such as using the infinite instead of present tense), split compounds (such as "kycklinglever" (chicken liver) as opposed to "kyckling lever" (chicken is alive)), word errors (such as word order and inserted or missing words) and style errors (such as constructs not conforming to language norms and missing commas).

An interesting observation is that the introduction of linguistic knowledge dramatically reduces the number of false detections. The number of correct detections were also decreased but not nearly to the same extent. For the first experiment at threshold 1.0, the detections decreased to 1/2 of the original count (from 59 to 30), while the false alarms decreased to 1/8 (from 78 to 10) of the unsupervised count. In the second test (threshold 4.0) the detections decreased to about 2/3 (from 122 to 78) while the false alarms reduced to about 1/6 (from 293 to 51) of the original count. The third test (threshold 8.0) also decreased the detections to 2/3 (from 148 to 100) and the false alarms dropped to 1/5 (from 428 to 94). From Table 2, we note that the precision is doubled or better in two of the three tests. The figures seem to suggest that the use of linguistic knowledge is promising in this context since it greatly reduces the false alarms compared to the decrease in correct detections.

Another aspect of probabilistic algorithms is how they relate to other algorithms. Most of

Table 2: *Comparison between probabilistic error detection methods. Methods in columns denoted* unsup *are unsupervised. Methods in columns denoted* phrase *make use of linguistic information. The* thr *figure is the ungrammaticality threshold for the weighted trigrams. The figures in the table are the number of detections while the figures in parentheses give the recall.*

Category	*thr*= 1.0 *unsup*	*thr*= 1.0 *phrase*	*thr*= 4.0 *unsup*	*thr*= 4.0 *phrase*	*thr*= 8.0 *unsup*	*thr*= 8.0 *phrase*
Spelling errors resulting in a non−existing word	25 (13%)	11 (5.5%)	50 (25%)	24 (12%)	59 (30%)	36 (18%)
Spelling errors resulting in an existing word	9 (14%)	5 (7.9%)	15 (24%)	9 (14%)	19 (30%)	10 (16%)
Erroneously split compound	14 (18%)	7 (9.2%)	30 (39%)	24 (32%)	37 (49%)	26 (34%)
Verb errors	2 (29%)	1 (14%)	4 (57%)	3 (43%)	5 (71%)	3 (43%)
Word errors	4 (10%)	1 (2.6%)	9 (23%)	6 (15%)	12 (30%)	10 (25%)
Style	5 (12%)	5 (12%)	14 (33%)	12 (29%)	16 (38%)	15 (36%)
Detected errors, total	59	30	122	78	148	100
False alarms	78	10	293	51	428	94
Precision	43%	75%	29%	60%	26%	52%

the error types discussed here are not comparable since other algorithms are not intended to detect them. For example, confusion set based methods (e.g. Yarowsky (1994), Golding (1995), Golding and Roth (1996), Mays et al. (1991), Golding and Schabes (1996), Mangu and Brill (1997)) is mainly concerned with competence spelling errors. Thus, comparing detection of performance spelling errors would be unjust. We have not yet conducted comparative tests on competence spelling errors (i.e. confusion sets type errors), but are confident that the above-mentioned methods are better suited for this task.

On the other hand, the comparison between the methods proposed here and rule-based error detection systems (see e.g. Jensen et al. (1983)) may be of relevance. Preliminary tests with a rule-based grammar checker for Swedish seem to indicate that the overlap in detection is limited, which is positive. This may suggest that they can be used in a complementary manner.

5 Future Work

Representatives are not restricted to part-of-speech tags. We will consider adopting this approach to representing phrase n-grams. This would enlarge the scope of the local error detection algorithms and give it a structural overview.

We will also consider methods for classification of the detections. This is necessary for at least two reasons. First, a user-friendly system should provide some sort of diagnosis for the detections made. Second, if the error is successfully classified, it may facilitate the generation of correction suggestions. Furthermore, existing correction methods may be applied to some error types.

6 Conclusions

In this paper we have presented two experiments with a robust probabilistic method for detecting context-sensitive spelling errors. Results show that the unsupervised method has a good error detection capacity but also a high rate of false alarms. By combining the unsupervised method with robust phrase and clause recognition we can significantly improve the unsupervised method.

References

J. Bigert. 2002. Automatic extraction of POS tag distance metrics by using n-grams. *Manuscript.*

J. Carlberger and V. Kann. 1999. Implementing an efficient part-of-speech tagger. *Software–Practice and experience*, 29(9):815–832.

R. Domeij, O. Knutsson, J. Carlberger, and V. Kann. 2000. Granska – an efficient hybrid system for Swedish grammar checking.

In T. Nordgård, editor, *Nodalida '99 Proceedings from the 12th Nordiske datalingvistikkdager*, pages 28–40. Department of Linguistics, University of Trondheim.

E. Ejerhed, G. Källgren, O. Wennstedt, and M. Åström. 1992. *The Linguistic Annotation System of the Stockholm-Umeå Project*. Department of Linguistics, University of Umeå.

E. Ejerhed. 1999. Finite state segmentation of discourse into clauses. In A. Kornai, editor, *Extended Finite State Models of Language*, chapter 13. Cambridge University Press.

A. Golding and D. Roth. 1996. Applying winnow to context-sensitive spelling correction. In *International Conference on Machine Learning*, pages 182–190.

A. Golding and Y. Schabes. 1996. Combining trigram-based and feature-based methods for context-sensitive spelling correction. In Arivind Joshi and Martha Palmer, editors, *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, pages 71–78, San Francisco. Morgan Kaufmann Publishers.

A. Golding. 1995. A Bayesian hybrid method for context-sensitive spelling correction. In David Yarovsky and Kenneth Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 39–53, Somerset, New Jersey. Association for Computational Linguistics.

H. Hahne. forthcoming. Utvärdering av en satssegmenterare (in Swedish) (Evaluation of a clause segmenter for Swedish). Master's thesis, Department of Linguistics, Uppsala University.

K. Jensen, G. Heidorn, L. Miller, and L. Ravin. 1983. Parse fitting and prose fixing: getting a hold on ill-formedness. *American Journal of Computational Linguistics*, 9(3–4):147–160.

V. Johansson. 2000. NP-detektion – utvärdering och förslag till förbättring av Granskas NP-regler (in Swedish) (NP-detection – evaluation and suggestions for improvement of Granska's NP rules), Bachelor's thesis, Department of Linguistics, Stockholm University.

F. Karlsson, A. Voutilainen, J. Heikkilä, and A. Anttila. 1995. *Constraint Grammar. A Language Independent System for Parsing Unrestricted text*. Mouton de Gruyter, Berlin, Germany.

O. Knutsson, J. Bigert, and V. Kann. forthcoming. Glass box evaluation of a robust shallow parser for Swedish.

L. Mangu and E. Brill. 1997. Automatic rule acquisition for spelling correction. In *Proc. 14th International Conference on Machine Learning*, pages 187–194. Morgan Kaufmann.

E. Mays, F. Damerau, and R. Mercer. 1991. Context based spelling correction. *Information Processing and Management*, 27(5):517–522.

D. Yarowsky. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Meeting of the Association for Computational Linguistics*, pages 88–95.