Number Sense Disambiguation

Stuart Moore
Computer Laboratory
University of Cambridge
stuart.moore
@cl.cam.ac.uk

Anna Korhonen
Computer Laboratory
University of Cambridge
anna.korhonen
@cl.cam.ac.uk

Sabine Buchholz
Cambridge Research Laboratory
Toshiba Research Europe
sabine.buchholz
@crl.toshiba.co.uk

Abstract

Word Sense Disambiguation is a well studied field, with a range of successful methods. However, there has been little work on examining the analogue for numbers, classifying them into senses ('year', 'date', 'telephone number' etc.) based on their context - potentially useful for Text to Speech and Information Extraction systems. We extend the semi-supervised Decision List model described by David Yarowsky (1994), bringing the model to a problem on which little work has been done. We report promising results and present a thorough error analysis which highlights several areas where the current methodology needs to be extended to deal better with number senses. We conclude by proposing several directions for future work.

1 Introduction

Numerical digits in text can be used to express a variety of different meanings. For example, *'2008'* could mean a year (*'Spain won the 2008 European Championships'*), a time (*'Our train arrives at 2008'*), a quantity (*'There were 2008 sweets in the jar'*) or a telephone number (*'Call 0800 508 2008 for more details'*). Pronunciation can vary according to the context - e.g. *'1990'* would be pronounced *'nineteen ninety'* as a year, but *'one thousand, nine hundred and ninety'* as a quantity. Determining the meaning for a given context is in many ways similar to Word Sense Disambiguation (WSD), but there are key differences. No dictionaries or resources equivalent to Wordnet exist, to allow us to exhaustively list the possible meanings.

Since there are an infinite number of numbers, building one model for each individual digit string would be inappropriate, as inevitably there would be unseen examples in the test data. Intuitively, replacing *'1998'* with *'1999'* in a sentence would not effect the (grammatical) correctness of the sentence, so it seems appropriate that the two numbers share a model. However, one would expect *'1'* or *'1999.5'* to have a different model, since they would occur in different contexts.

For some tasks, for example Text to Speech, a system that mapped numbers into their expanded word form (*'1998'* to *'nineteen ninety eight'* or *'one thousand, nine hundred and ninety eight'* as appropriate) would be sufficient. However, we have decided to split the problem into two: classifying the number, and then expanding the classified number. There are several reasons why we adopt this approach. Firstly, assigning a number a sense is useful for NLP tasks such as information extraction systems and parsers. Secondly, the expansion is dialect dependent; for example the year *'2008'* is generally expanded to *'two thousand **and** eight'* by British-English speakers, but *'two thousand eight'* by (at least some dialects of) American-English speakers. It seems more practical to have a dialect independent disambiguation system, and an expansion system can vary with dialect as appropriate.

Thirdly, the method of expanding years varies according to the year (compare *'1999'* — *'nineteen ninety nine'*; *'2000'* — *'two thousand'*; *'2010'* — *'two thousand and ten'* or *'twenty ten'*) but the underlying features indicating that they are years and not another number sense should not vary significantly. By concentrating on expansions, two different models for years have to be found, and therefore each will have less train-

ing data. This applies equally to other numbers which have several possible expansions, for example telephone numbers or times.

Using the labelled corpus of Sproat et al. (2001) we extend the semi-supervised Decision List model described by David Yarowsky (1995) to this problem on which little work has been done.

Our results and thorough error analysis reveal several areas which require improvement until success can be obtained in this task. These concern the number categories and labels in the corpus we used, as well as the extended semi-supervised method which needs further improvement to deal with the full range of number senses.

2 Related Work

Sproat et al. (2001) describe a supervised system designed to normalise any 'Non Standard Words' (NSWs) - 'words' such as abbreviations, acronyms and numbers which have more complicated pronunciation conventions than normal words, and so need special handling by Text to Speech systems.[1] For example, *'mph'* (*'miles per hour'*) and *'CDs'* (*'cee dees'*) are NSWs. This included defining a list of number senses (shown in Table 1) and training their system to classify numbers into these senses, although this was not their main focus. We have used their corpus data for our experiments, and adopt their number classifications. They labelled four corpora in different domains using this classification system, the most extensive being a subset of the North American News Text Corpus (NANTC) (Graff, 1995).

As the table shows, there is a huge variation in the frequency of different categories. These frequencies are also going to vary with different domains: in newspapers it is rare to come across telephone numbers or addresses, whereas amounts of money, years and dates are relatively frequent; whereas in email, telephone numbers and addresses are likely to be more common.[2]

Their system performs classifications using a decision tree system to form a lattice, and a trigram language model to make the final choice. The decision tree uses 136 features, some of which are domain dependent, based on the target token and 2 tokens to either side. Features include the alphanumeric content (alphabetic, numeric or mixture), vowel/consonant content, casing (all upper, lower or mixed) and punctuation symbols. They also build a model for each domain to differentiate between three of the non-numeric classes; results from this model are added to the feature list used to build the main model decision tree (Sproat et al., 2001, Section 6.4.2). These features are geared towards expanding alphabetical tokens (although many are useful for numbers as well); the authors do not specify whether any other features relating to numbers are used. The trigram language model makes the final choice between different classifications provided by the decision tree. Numbers are converted to their numeric tags (*NYER*, *NTIME* etc.) to train the language model, and we believe the language model plays a large part in their performance on Number Sense Disambiguation (NSD). They evaluate their system as a whole, not on numbers specifically. We took the model they trained using the NANTC, and evaluated it on an unseen part of that corpus. The system achieves 97.6% accuracy on the numeric tags.

Label		Description	Examples
NUM	(57%)	Number (Cardinal)	12, 45, 1/2, 0.6
NYER	(20%)	Year(s)	1998, 80s, 1900s
NORD	(8.7%)	Number (Ordinal)	May 7, 3rd, Bill Gates III
MONEY	(7.7%)	Money (US or other)	$3.45, HK$300, Y20,000, $200K
NIDE	(2.7%)	Identifier	747, 386, I5, pc110
NTEL	(1.4%)	Telephone number	212 555–4523
NTIME	(1.2%)	a (compound) time	3:20, 11:45
NDATE	(0.82%)	a (compound) date	2/2/99, 14/03/87 (or US) 03/14/87
NDIG	(0.20%)	Number as digits	Room 101
NADDR	(0.18%)	Building Number	5000 Pennsylvania, 28 Kings Parade
NZIP	(0.18%)	Zip code or PO box	91020
PRCT	(0.06%)	Percentage	75%, 3.4%

Table 1: Number Categories. (Percentages indicate frequency in NANTC.)

Number Sense Disambiguation also shares some qualities with Named Entity Recognition, such as described in the CoNLL shared tasks (Tjong Kim Sang and De Meulder, 2003). Both require classifying a word (or a few adjacent words) into one of a number of categories based on the context; and in both cases, the majority of the surrounding context does not require classification. The Named Entity task of the Seventh Message Understanding Conference (Chinchor, 1998) included identifying times, dates, amounts of money and percentages (although, again, this was not the main aim of the task). Unlike Sproat et al. they include identifying numbers written as words within text (*'several million dollars'*), and also more abstract dates and times (*'July last year'*), meaning their task has a slightly different aim. The system

[1] There is an unsupervised component, but it does not relate directly to number senses. (Sproat et al., 2001, Sec. 7)

[2] Note that Sproat et al. label a day of the month - e.g. *'July 13'* - as an ordinal, NORD. The date category, NDATE, is reserved for compound dates such as *'2/2/89'*.

of Mikheev et al. (1998) uses a rule-based system to identify the numbers, because *"temporal and numeric expressions within English newspapers have a fairly structured appearance."* They achieve a precision of 98% and a recall of 89% over the different number types, suggesting that grammar based methods are a solution too - although the rules are likely to be domain specific, and rule based systems may not work as well for less regular domains.

Yarowsky (1994) describes a decision list approach to WSD problems. Since this method is well understood and evaluated in WSD, we apply it to NSD with a number of extensions. This enables us to best investigate the specific challenges NSD poses for existing WSD methodology and NLP in general, as well as allowing us to identify where further research is needed.

Yarowsky's approach is as follows. During training, a series of rules are formed. Rules can be based on a word or stem occurring within a $\pm k$ window of the target word, the word immediately before or after the target word, or the pair of words before, after, or either side of the target word.

The system counts the number of examples of each class which match a rule, and calculates the log likelihood of that rule indicating the most common class. Let S be the set of sentences matching a rule, and A being those sentences with label A.

$$\text{LogLike} = -\log \frac{\mathbb{P}(s \in A | s \in S)}{\mathbb{P}(s \notin A | s \in S)}$$

Where the probabilities are estimated thus:

$$\mathbb{P}(s \in A | s \in S) = \frac{|A \cap S|}{|S| + \alpha}$$

$$\mathbb{P}(s \notin A | s \in S) = \frac{|\overline{A} \cap S| + \alpha}{|S| + \alpha}$$

Here α is a parameter used to deal with the situation where a rule has no counter examples. We use $\alpha = 0.5$; experiments with other values showed little variation.

Rules are then ranked according to their log likelihood. When evaluating a target word against the model, the class of the highest ranked rule that matches its context is used. Yarowsky provides evidence that using a single rule in this fashion outperforms combining multiple rules for this task.

A cut-off log likelihood is set. If none of the rules above this cut-off point have matched the target word, either the system leaves it unlabelled or it provides a default label.

Yarowsky (1996) also uses this method to perform three number classification tasks. The system is tested comparing fractions and dates of the form 3/4 (with 94% accuracy), years and quantities (93%), and whether Roman numerals should be pronounced as cardinals or ordinals (97%).

The work discussed so far is fully supervised. Yarowsky (1995) also presents a semi-supervised version, used for standard WSD.[3] He providing a few labelled 'seed' examples, training a model on them and applying it to a large unlabelled corpus. The resulting labelled sentences are then used to train a new model. This process is repeated, with the output from each iteration providing the input for the new iteration. The method achieves an average performance of 90.6% accuracy, comparable to a supervised algorithm on the same data.

3 Our Approach

We adopt the semi-supervised version of Yarowsky's method, since this carries the greatest potential for economically adapting models to different domains where the number senses may vary to a large degree. However, we introduce a number of extensions which enable dealing with numbers better: features relating to punctuation and numbers, and the allowable combinations of features. Yarowsky's method has mainly been used for dual class classifications, but naturally generalises to a multi-class problem. All experiments use the subset of the NANTC used by Sproat et al. The training set used contained 31,597 numbers, and the test set 12,003.

The feature set we use is a variant of that used by Yarowsky. He uses word tokens, their Part of Speech (POS) tags (which we do not use)[4] and word stems (Yarowsky, 1994). We additionally include features to capture punctuation, and some number specific features:

- The token with all digits removed, converted to lower case (i.e. '1st' would produce 'st').
- The digits of the token (removing all other characters)
- The number of digits in the token; this can be used to identify potential years and zip codes.
- The number rounded to one significant figure (so '1990', '1570', '2024' all map to '2000') — useful for capturing classes of numbers that tend to be in certain regions (e.g. years in text are likely to be near 2000).
- Whether the number had leading zeros (e.g. 0845 and 00012345 have leading zeros but 0.42 does not).

All features are identified at a specific location relative to the target token, examining 5 tokens to

[3] Yarowsky uses the term unsupervised, however we use the term semi-supervised because some labelled examples have to be provided when training the system.

[4] We do not use POS tags in order to investigate how well number senses can be identified independently of other tools. Future work may include adding POS tags to the feature set.

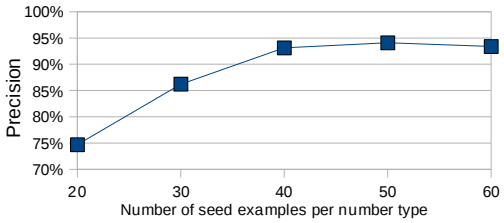


Figure 1: First Iteration

either side. This includes features of the target token itself. This is an extension on Yarowsky's method, which only considers words or word pairs immediately adjacent to the target token; there are some senses, for example addresses and postal codes, where these extra tokens could be useful.

Additionally, we examine the wider context. Following his method, we list all features occurring anywhere within a window centred on the target number, using windows of $\pm$ 5 and 15 tokens.[5] Initially we did this for all features, but often numbers or punctuation within the wider context lead to poor performance, so we now only record word features within these windows.

We also consider compound features by taking pairs of the simple features described above. A number matches the compound feature if it has both of the simple features. This allows the bigrams used by Yarowsky, but is more powerful in that it also allows other combinations of features. For example, *'in'* as the word before a number, combined with *'when'* within the $\pm$ 15 token window, was found to be a stronger indicator that the target number was a year than the presence of *'in'* alone. We rank the features by calculating the log likelihood as described in section 2.

3.1 First Iteration

We seek a high precision first iteration, as a good basis for future iterations; we accept poor recall to further this aim. The following strategy was found to provide the highest precision: Take the three highest ranked rules for each sense (more in the event of a tie), and apply these to the entire (unlabelled) training set. If two conflicting rules match the same sentence in the training set, it remains unlabelled (rather than letting the highest scoring rule take precedence).[6]

[5] We additionally investigated adding a window of $\pm$ 10 tokens, but this did not improve performance.

[6] This is a departure from Yarowsky's method. We initially followed Yarowsky's method, but found from experiments on the training data that in the first iteration poor rules for some classes were hiding good rules from others, which happened to score lower due to the small number of training examples.

We investigated the number of seed examples needed for a high precision first iteration. n seed examples of each category were randomly selected to be the labelled data. Figure 1 shows the effect of varying n on the precision of the first iteration. The chart shows that adding seed examples until there are 40 in each category is beneficial (achieving 93.1% precision on the remainder of the unlabelled training data), but after this point there is limited improvement. Recall in all cases was in the region 27%-33%. We use 40 seed examples per category for all future experiments.

3.2 Second Iteration

We perform a second iteration of the system, assigning categories as with Yarowsky's original method. Figure 2(a) shows the performance against the system's training data, and 2(b) over unseen test data.

For the training data, taking the default class into account gives the optimal cutoff at 5.0, leading to an accuracy of 84%. Applying this cutoff to the test set gives 74.9% accuracy; this is very close to the optimum cutoff on the test data (5.2 - 75.2% accuracy), suggesting that a cutoff chosen based on one data set will perform well on others.

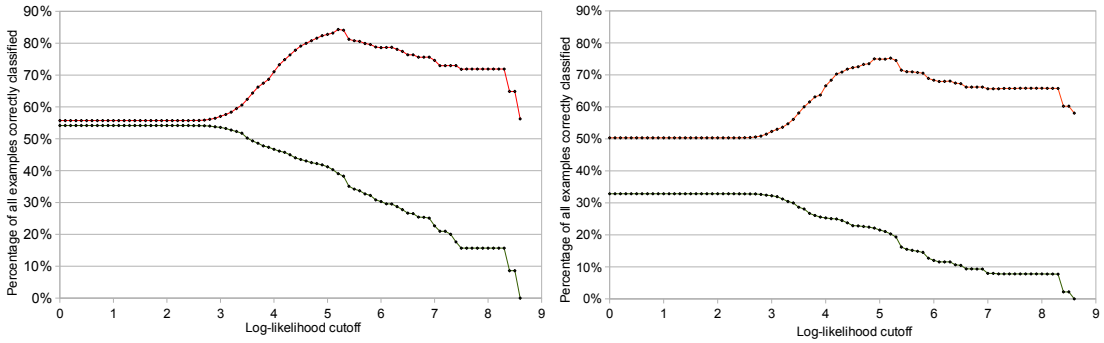
The peak performance of 75% is some way off that of Sproat et al. but it should be remembered that theirs had 31,597 labelled training examples, compared to our 480 labelled and 31,117 unlabelled. The baseline performance is 58.0% on the test data (based on labelling everything as NUM, the most frequent category).

Yarowsky obtains best performance by iterating his system a number of times, however running further iterations on our system leads to a drop in performance. We hope that addressing the issues described in the next section will lead to an improvement of the first and second iterations, allowing further iterations to be helpful.

4 Error Analysis

4.1 First Iteration

We analysed 100 misclassifications. The most common issue was numbers incorrectly being labelled as ordinals because they were preceded by a full stop. The seed examples contained several sentences such as *'The Federal Reserve might have to be more aggressive in raising interest rates at its meeting on Nov. 15'*.



(a) The performance of the system on its training data

(b) The performance of the system on the test data

Figure 2: Second Iteration. In each graph, the x axis represents our ability to choose a log-likelihood cut off point - only rules with log likelihood scores above this value will be used when classifying the data. The lower line represents the proportion of all examples that this system correctly labels. The upper one represents how the performance is affected by taking all the data remaining unlabelled after the second iteration, and assigning it the category NUM (the most common).

This lead to a rule that a two digit number following a full stop was identified as a day of the month. However, this leads to a misclassification of 35% of examples, such as *'placing it at No. 30 on Forbes' recent ranking'* where the token before the number happens to end in a full stop. Twenty of these were apparently table data in the original newspaper articles, where the data made little sense without the tabular formatting, e.g. *'John Updike. .. 10 ... - ... 2 (Knopf, $25.95.) '*

We can add a list of months and their abbreviations as a new feature to avoid these errors. Months are domain independent and equivalent lists can be easily generated for other languages.

The next most common error (17%) was from numbers followed by a colon being identified as times. All seed example times were of the form 20:00, and so the system correctly identified that the colon was an indicator of the NTIME class; however at present there are no suitable features to capture the difference between a colon in the middle of or at the end of a token, meaning all numbers followed by a colon were misclassified.

10% of errors were due to the target number being adjacent to another four digit number, and labelled a year. This rule originated from examples such as *'November 1, 1994'*, but caused misclassifications such as *'United States v. Cochran, 883 F. 2d 1012, 1017-1018 (CA11 1989) ;'* where a complex reference is mislabelled as an ordinal. These errors were restricted to references, as above, or from tables of financial data. We hope to identify these and treat them separately.

8% of errors were due to corpus mislabelling; the system had chosen the correct category (NORD for a day of the month) but the corpus had been incorrectly labelled. The remainder of the errors were caused by rules picking up on words that were not useful, often from article metadata (e.g. *'Times'* - from *'New York Times'* - was associated with the NDATE class because of standard metadata format). Metadata was left in to enable a fair comparison with the Sproat et al. system, but could be removed for future experiments.

Common words (e.g. *'the'*, *'that'*) frequently appeared in rules, often confusing the system. These can be removed using a frequency filter.

More complicated feature combinations (i.e. a rule combining three or four features) will be needed to discover some of the more complicated contexts. There are too many features for a 'brute force' method to work, so some other method of feature selection will be needed.

The classification categories may not be the ideal ones for the problem, for example creating a 'day of month' category to distinguish them from other ordinal numbers would allow the system to model their separate characteristics.

Some of the corpus was labelled inconsistently. In particular, many percentages had been incorrectly labelled NUM (general number), and the distinction in usage between NIDE (Identifier) and NDIG (Number as digits) is not always clear.

4.2 Second Iteration

Again, we analysed 100 incorrectly classified examples. Many of the errors seen in the first iteration were repeated in the second iteration. In-

correct interpretation of punctuation led to 24% of the errors - mainly because the system could not differentiate between punctuation within a number and punctuation immediately after it. 13% of errors relate to the presence of double dashes, which mainly occur within article metadata, and 12% were due to full stops preceding the number.

The most common error not seen in the first iteration (10%) was the system assuming that *'in'* preceding a number indicated that it was a year, learnt from examples such as *'The reduction in the deficit from $290 billion in 1992...'*. This lead to misclassifications in sentences such as *'The landslide opposition victory in 15 of Serbia's 18 biggest towns...'*. In this case, there is a higher scoring rule matching most of the positive examples - combining the presence of *'in'* with the fact that the target number has four digits. Therefore, the rule that leads to the misclassifications doesn't actually lead to any correct classifications. We intend to explore methods of identifying rules where this is the case, and removing them from the final ruleset.

In summary, our error analysis reveals a number of problems which require addressing before high accuracy NSD can be achieved. These concern the number categories, the labelled corpus, the way linguistic constructions are handled, as well as the semi-supervised technique itself. The error analysis provides us with a framework for further development of the approach.

5 Conclusion

There has been little work dealing with Number Sense Disambiguation, despite the fact that much work has been done on the related problem of WSD, and models of NSD could be useful for many real-world systems, from Text to Speech to Information Extraction. Our study has taken the well-known semi-supervised Decision List model for WSD described by Yarowsky (1995) and investigated the applicability of the model to NSD. We obtained encouraging results, but our error analysis revealed a number of areas which require further work until this method originating from WSD can be successfully applied to NSD.

Another method of improving the performance of the second (and subsequent) iterations would be to investigate whether the concept of 'one sense per discourse' - the observation that most ambiguous words only occur in one sense within a single document (Yarowsky, 1995) - has an analogue for number sense disambiguation. Most documents contain several number senses but it might be that there are analogues when certain features are taken into account. For example, if a document contains many years, can we assume that any remaining unclassified four digit numbers within the document are also years?

There is also a natural extension to investigate the performance of other semi- and minimally-supervised methods, and to explore whether these semi-supervised techniques have an advantage over fully supervised techniques or grammar rule based systems when adapting to a new domain.

Acknowledgements

Stuart Moore was supported by the EPSRC CASE Studentship co-sponsored by Toshiba Research Europe. We would like to thank Prof. Alan Black of Carnegie Mellon University for providing the corpora labelled by Sproat et al. (2001)

References

Nancy A. Chinchor. 1998. Overview of MUC-7/MET-2. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*.

David Graff. 1995. North American News Text Corpus. Linguistic Data Consortium.

Andrei Mikheev, Claire Grover, and Marc Moens. 1998. Description of the LTG system used for MUC-7. In *Proceedings of the Seventh Message Understanding Conference (MUC-7)*.

Richard Sproat, Alan W. Black, Stanley F. Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. 2001. Normalization of non-standard words. *Computer Speech & Language*, 15(3) pages 287–333.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of CoNLL-2003*.

David Yarowsky. 1994. Decision lists for lexical ambiguity resolution: Application to accent restoration in Spanish and French. In *Proceedings of the 32nd Meeting of the ACL*.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd Meeting of the ACL*.

David Yarowsky. 1996. Homograph disambiguation in text-to-speech synthesis. In *Progress in Speech Synthesis*, Springer-Verlag, New York.