Language Modeling for Determiner Selection

Jenine Turner and Eugene Charniak
Department of Computer Science
Brown Laboratory for Linguistic Information Processing (BLLIP)
Brown University
Providence, RI 02912
{jenine|ec}@cs.brown.edu

Abstract

We present a method for automatic determiner selection, based on an existing language model. We train on the Penn Treebank and also use additional data from the North American News Text Corpus. Our results are a significant improvement over previous best.

1 Introduction

Determiner placement (choosing if a noun phrase needs a determiner, and if so, which one) is a non-trivial problem in several language processing tasks. While context beyond that of the current sentence can sometimes be necessary, native speakers of languages with determiners can select determiners quite well for most NPs. Native speakers of languages without determiners have a much more difficult time.

Automating determiner selection is helpful in several applications. A determiner selection program can aid in Machine Translation of determiner-free languages (by adding determiners after the text has been translated), correct English text written by non-native speakers (Lee, 2004), and choose determiners for text generation programs.

Early work on determiner selection focuses on rule-based systems (Gawronska, 1990; Murata and Nagao, 1993; Bond and Ogura, 1994; Heine, 1998). Knight and Chander (1994) use decision trees to choose between *the* and *a/an*, ignoring NPs with no determiner, and achieve 78% accuracy on their Wall Street Journal corpus. (Deciding between *a* and *an* is a trivial postprocessing step.)

Minnen et al. (2000) use a memory-based learner (Daelemans et al., 2000) to choose determiners of base noun phrases. They choose between no determiner (hencefore *null*), *the*, and *a/an*. They use syntactic features (head of the NP, part-of-speech tag of the head of the NP, functional tag of the head of the NP, category of the constituent embedding the NP, and functional tag of the constituent embedding the NP), whether the head is a mass or count noun and semantic classes of the head of the NP (Ikehara et al., 1991). They report 83.58% accuracy.

In this paper, we use the Charniak language model (Charniak, 2001) for determiner selection. Our approach significantly improves upon the work of Minnen et al. (2000). We also use additional automatically parsed data from the North American News Text Corpus (Graff, 1995), further improving our results.

2 The Immediate-Head Parsing Model

The language model we use is described in (Charniak, 2001). It is based upon a parser that, for a sentence s, tries to find the parse π defined as:

$$\arg max_{\pi} p(\pi \mid s) = \arg max_{\pi} p(\pi, s) \qquad (1)$$

The parser can be turned into a language model $p(s)$ describing the probability distribution over all possible strings s in the language, by considering all parses π of s:

$$p(s) = \sum_{\pi} p(\pi, s) \qquad (2)$$

Proceedings of NAACL HLT 2007, Companion Volume, pages 177–180,
Rochester, NY, April 2007.

Here $p(\pi, s)$ is zero if the yield of $\pi \neq s$.

The parsing model assigns a probability to a parse π by a top-down process. For each constituent c in π it first guesses the pre-terminal of c, $t(c)$ (t for "tag"), then the lexical head of c, $h(c)$, and then the expansion of c into further constituents $e(c)$. Thus the probability of a parse is given by the equation

$$\begin{aligned} p(\pi) &= \prod_{c \in \pi} p(t(c) \mid l(c), H(c)) \\ &\quad \cdot\ p(h(c) \mid t(c), l(c), H(c)) \\ &\quad \cdot\ p(e(c) \mid l(c), t(c), h(c), H(c)) \end{aligned}$$

where $l(c)$ is the label of c (e.g., whether it is a noun phrase NP, verb phrase, etc.) and $H(c)$ is the relevant history of c — information outside c deemed important in determining the probability in question. $H(c)$ approximately consists of the label, head, and head-part-of-speech for the parent of c: $m(c), i(c)$, and $u(c)$ respectively and also a secondary head (e.g., in "Monday Night Football" Monday would be conditioned on both the head of the noun-phrase "Football" and the secondary head "Night").

It is usually clear to which constituent we are referring and we omit the (c) in, e.g., $h(c)$. In this notation the above equation takes the following form:

$$\begin{aligned} p(\pi) &= \prod_{c \in \pi} p(t \mid l, m, u, i) \cdot p(h \mid t, l, m, u, i) \\ &\quad \cdot\ p(e \mid l, t, h, m, u). \end{aligned} \tag{3}$$

Next we describe how we assign a probability to the expansion e of a constituent. We break up a traditional probabilistic context-free grammar (PCFG) rule into a left-hand side with a label $l(c)$ drawn from the non-terminal symbols of our grammar, and a right-hand side that is a sequence of one or more such symbols. For each expansion we distinguish one of the right-hand side labels as the "middle" or "head" symbol $M(c)$. $M(c)$ is the constituent from which the head lexical item h is obtained according to deterministic rules that pick the head of a constituent from among the heads of its children. To the left of M is a sequence of one or more left labels $L_i(c)$ including the special termination symbol $\triangle$, which indicates that there are no more symbols to the left. We do the same for the labels to the right, $R_i(c)$. Thus, an expansion $e(c)$ looks like:

$$l \rightarrow \triangle L_m ... L_1 M R_1 ... R_n \triangle. \tag{4}$$

The expansion is generated first by guessing M, then in order L_1 through L_{m+1} (= $\triangle$), and then, R_1 through R_{n+1}.

Let us turn to how this works in the case of determiner recovery. Consider a noun-phrase, which, missing a possible determiner, is simply "FBI." The language model is interested in the probability of the strings "the FBI," "a/an FBI" and "FBI." The version with the highest probability will dictate the determiner, or lack thereof. So, consider (most of) the probability calculation for the answer "the FBI:"

$$\begin{aligned} & p(\text{NNP} \mid H) \cdot p(\text{FBI} \mid \text{NNP}, H) \\ &\quad \cdot\ p(\text{det} \mid \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\triangle \mid \text{det}, \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\text{the} \mid \text{det}, \text{FBI}, \text{NNP}, H) \end{aligned} \tag{5}$$

Of these, the first two terms, the probability that the head will be an NNP (a singular proper noun) and the probability that it will be "FBI", are shared by all three competitors, *null*, *the*, and *a/an*. These terms can therefore be ignored when we only wish to identify the competitor with the highest probability. The next two probabilities state that the noun-phrase contains a determiner to the left of "FBI" and that the determiner is the last constituent of the left-hand side. The last of the probabilities states that the determiner in question is *the*. Ignoring the first two probabilities, the critical probabilities for "the FBI" are:

$$\begin{aligned} & p(\text{det} \mid \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\triangle \mid \text{det}, \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\text{the} \mid \text{det}, \text{FBI}, \text{NNP}, H) \end{aligned} \tag{6}$$

Conversely, to evaluate the probability of the noun-phrase "FBI" — i.e., no determiner, we evaluate:

$$p(\triangle \mid \text{FBI}, \text{NNP}, H) \tag{7}$$

We ask the probability of the NP stopping immediately to the left of "FBI." For "a/an FBI" we evaluate:

$$\begin{aligned} & p(\text{det} \mid \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ p(\triangle \mid \text{det}, \text{FBI}, \text{NNP}, H) \\ &\quad \cdot\ (p(\text{a} \mid \text{det}, \text{FBI}, \text{NNP}, H) + \\ &\quad\quad p(\text{an} \mid \text{det}, \text{FBI}, \text{NNP}, H)) \end{aligned} \tag{8}$$

Test Data	Method	Accuracy
leave-one-out	Minnen et al.	83.58%
	Language Model (LM)	86.74%
tenfold on development	LM	84.72%
	LM trained on WSJ + 3 million words of NANC	85.83%
	LM trained on WSJ + 10 million words of NANC	86.36%
	LM trained on WSJ + 20 million words of NANC	86.64%
tenfold on test	LM trained on WSJ + 20 million words of NANC	86.63%

Table 1: Results of classification

This equation is very similar to Equation 6 (the equation for “the FBI”, except the term for the probability of *the* is replaced by the sum of the probabilities for *a* and *an*.

To choose between *null*, *the*, or *a/an*, the language model in effect constructs Equations 6, 7 and 8 and we pick the one that has the highest probability.

2.1 Training the model

As with (Minnen et al., 2000), we train the language model on the Penn Treebank (Marcus et al., 1993). As far as we know, language modeling always improves with additional training data, so we add data from the North American News Text Corpus (NANC) (Graff, 1995) automatically parsed with the Charniak parser (McClosky et al., 2006) to train our language model on up to 20 million additional words.

3 Results and Discussion

The best results of Minnen et al. (2000) are using leave-one-out cross-validation. We also test our language model using leave-one-out cross-validation on the Penn Treebank (Marcus et al., 1993) (WSJ), giving us 86.74% accuracy (see Table 1).

Leave-one-out cross-validation does not make sense in this case. When choosing determiners, we can train a language model on similar data, but not on other NPs in the article. Therefore, for the rest of our tests, we use tenfold cross-validation. The difference between leave-one-out and tenfold cross-validation is due to the co-occurrence of NPs within an article. Church (2000) shows that a word appears with much higher probability when seen elsewhere in an article. Thus, a rare NP might be unseen in tenfold cross-validation, but seen in leave-one-out.

For each of our sets in tenfold cross validation, we use 80% of the Penn Treebank for training, 10% for development, and 10% for testing. The divisions occur at article boundaries. On our development set with tenfold cross-validation, we get 84.72% accuracy using the language model (Table 1).

As expected, we achieve significant improvement when adding NANC data over training on data from the Penn Treebank alone (Table 1). With 20 million additional words, we seem to be approaching an upper bound on the language model features. We obtain improvement despite the fact that the parses were automatic, but there may have been errors in determiner selection due to parsing error.

Table 2 gives “error” examples. Some errors are wrong (either grammatically or yielding a significantly different interpretation), but some “incorrect” answers are reasonable possibilities. Furthermore, even all the text of the article is not enough for classification at times. In particular note Example 5, where unless you know whether IBM was *the* world leader or simply one of the world leaders at the time of the article, no additional context would help.

4 Conclusions and Future Work

With the Charniak (Charniak, 2001) language model, our results exceed those of the previous best (Minnen et al., 2000) on the determiner selection task. This shows the benefits of the language model features in determining the most grammatical determiner to use in a noun phrase. Such a language model looks at much of the structure in individual sentences, but there may be additional features that could improve performance. There is a high rate of ambiguity for many of the misclassified sentences.

The success of using a state-of-the-art language

Guess	Correct	Sentence
the	*null*	(1) The computers were crude by **today's** standards.
null	*the*	(2) In addition, **the Apple II** was an affordable $1,298. (3) Highway officials insist **the ornamental railings** on older bridges aren't strong enough to prevent vehicles from crashing through.
a/an	*the*	(4) **The new carrier** can tote as many as four cups at once. (5) IBM, **the world leader** in computers, didn't offer its first PC until August 1981 as many other companies entered the market.
the	*a/an*	(6) In addition, the Apple II was **an affordable $1,298**. (7) "The primary purpose of **a railing** is to contain a vehicle and not to provide a scenic view," says Jack White, a planner with the Indiana Highway Department.
a/an	*null*	(8) Crude as they were, these early PCs triggered **explosive product development** in desktop models for the home and office.

Table 2: Examples of "errors"

model in determiner selection also suggests that one would be helpful in making other decisions in the surface realization stage of text generation. This is an avenue worth exploring.

Acknowledgements

This work was supported by NSF PIRE grant OISE-0530118. We would also like to thank the BLLIP team for their comments.

References

Francis Bond and Kentaro Ogura. 1994. Countability and number in Japanese-to-English machine translation. In *15th International Conference on Computational Linguistics*, pages 32–38.

Eugene Charniak. 2001. Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*. The Association for Computational Linguistics.

Kenneth Church. 2000. Empirical estimates of adaptation: The chance of Two Noriegas is closer to $p/2$ than p^2. In *Proceedings of COLING-2000*.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2000. TiMBL: Tilburg memory based learner, version 3.0, reference guide. ILK Technical Report ILK-0001, ILK, Tilburg University, The Netherlands.

Barbara Gawronska. 1990. Translation great problem. In *Proceedings of the 13th International Conference on Computational Linguistics*.

David Graff. 1995. *North American News Text Corpus*. Linguistic Data Consortium. LDC95T21.

Julia E. Heine. 1998. Definiteness predictions for Japanese noun phrases. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics*, pages 519–525.

Satoru Ikehara, Satoship Shirai, Akio Yokoo, and Hiromi Nakaiwa. 1991. Toward an MT system without preediting - effects of new methods in ALT-J/E. In *Third Machine Translation Summit*, pages 101–106.

Kevin Knight and Ishwar Chander. 1994. Automated postediting of documents. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 779–784.

John Lee. 2004. Automatic article restoration. In *Proceedings of the 2004 NAACL Conference Student Research Workshop*, pages 195–200.

Michell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of HLT-NAACL 2006*.

Guido Minnen, Francis Bond, and Ann Copestake. 2000. Memory-based learning for article generation. In *Proceedings of the Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop*, pages 43–48.

Masaki Murata and Makoto Nagao. 1993. Determination of referential property and number of nouns in Japanese sentences for machine translation into English. In *Fifth International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 218–225.