THE ROBUSTNESS OF AN ALMOST-PARSING LANGUAGE MODEL GIVEN ERRORFUL TRAINING DATA

Wen Wang[1,2], *Mary P. Harper*[1], *Andreas Stolcke*[2]

[1] Electrical and Computer Engineering, Purdue University
West Lafayette, IN 47907-1285
[2] Speech Technology and Research Laboratory, SRI International
Menlo Park, CA 94025
wwang@speech.sri.com, harper@ecn.purdue.edu, stolcke@speech.sri.com

ABSTRACT

An almost-parsing[1] language model has been developed [1] that provides a framework for tightly integrating multiple knowledge sources. Lexical features and syntactic constraints are integrated into a uniform linguistic structure (called a SuperARV) that is associated with words in the lexicon. The SuperARV language model has been found able to reduce perplexity and word error rate (WER) compared to trigram, part-of-speech-based, and parser-based language models on the DARPA Wall Street Journal (WSJ) CSR task. In this paper we further investigate the robustness of the language model to possibly inconsistent and flawed training data, as well as its ability to scale up to sophisticated LVCSR tasks by comparing performance on the DARPA WSJ and Hub4 (Broadcast News) CSR tasks.

1. INTRODUCTION

The purpose of a language model (LM) is to determine the *a priori* probability of a word sequence w_1^N, $P(w_1^N)$. Although word-based LMs (with bigram and trigram being the most common) remain the mainstay in many continuous speech recognition systems, recent efforts have explored a variety of ways to improve LM performance [2, 3, 4, 5]. There has been good progress in developing structured models [2, 3, 5] that focus more on the hierarchical characteristics of a language than specific information about words and their lexical features (e.g., case, number). Goodman [6] integrates a small set of features at the level of Context-free Grammar (CFG) production rules to achieve increased parse accuracy; however, only a small set of lexical features can be integrated without causing a significant increase in grammar size and a concomitant data sparsity problem. In contrast, we have developed an almost-parsing language model, called the SuperARV LM, that does not appear to suffer from this sparseness problem [1]. This structured LM tightly integrates structural constraints and lexical features at the word level by using a structure called a SuperARV. We investigate the SuperARV LM because it achieves a high level of word predictability with a complexity lower than a full parser-based LM and can be embedded in a decoder.

In contrast to word-based LMs which are trained using transcriptions, structured LMs need to be trained from parse annotations. Since there is only a limited availability of human annotated parse treebanks over a very limited number of tasks, developing a syntactically based LM for a new task can be quite difficult. One possibility is to use state-of-the-art parsers to generate a treebank for a new task; however, such a treebank would likely contain many incorrect parses. It is uncertain how sensitive various structured LMs are to errorful training data. Although Chelba [2] mentions that his structured LM should be able to recover from errors in an automatically generated treebank because of his use of EM parameter reestimation, this hypothesis is not explicitly tested. In this paper, we investigate the impact of errorful training data on our SuperARV LM. First we briefly review the SuperARV LM (Section 2) and then investigate its robustness and scale-up characteristics across two tasks (Section 3). Conclusions appear in Section 4.

2. BRIEF REVIEW OF THE SUPERARV LM

The SuperARV LM [1] is a highly lexicalized probabilistic LM based on Constraint Dependency Grammars (CDGs). It tightly integrates multiple knowledge sources, for example, word identity, lexical features, and syntactic and semantic constraints at both the *knowledge representation level* and *model level*.

The first type of integration was achieved by introducing a linguistic structure, called a super abstract role value (*SuperARV*), to encode multiple knowledge sources in a uniform representation that is much more fine-grained than part-of-speech (POS). A SuperARV is an abstraction of the joint assignment of dependencies for a word, which provides a mechanism for lexicalizing CDG parse rules. The gray box of Figure 1 presents an example of a SuperARV for the word *did*, which is derived from the dependency parse of the sentence *What did you learn* depicted in the white box of Figure 1. Each word in the parse has a lexical category, a set of feature values, and a governor role (denoted G) which is assigned a role value, comprised of a label, as well as a modifiee, which indicates the position of the word's governor or head. For example, the role value assigned to the governor role of *did* is `vp-1`, where its label `vp` indicates its grammatical function and its modifiee `1` is the position of its head *what*. The words in the parse can also have need roles (denoted *Need1*, *Need2*, and *Need3*), which are used to ensure the grammatical requirements (e.g., subcategorization) of a

This research was supported in part by Purdue Research Foundation, National Science Foundation under Grant No. BCS-9980054, and the Defense Advanced Research Projects Agency Information Awareness Office EARS program. ARPA Order No. N614, Program Code No. 2E20, issued by DARPA/CMO under Contract No. MDA972-02-C-0038. The Hub4 experiments were conducted by the first author at SRI International as a part of her doctoral dissertation. Part of this work was carried out while the second author was on leave at National Science Foundation. Approved for public release; distribution is unlimited.

[1] Almost-parsing involves obtaining structural information for an utterance without completely specifying the parse.

word are met. Note that the verb *did* needs a subject (*Need1*) and a base form verb (*Need2*), but since the word takes no other complements, the modifiee of the role value assigned to *Need3* is set equal to its own position.

A SuperARV is formally defined as a four-tuple for a word, $\langle C, F, (R, L, UC, MC)+, DC \rangle$, where C is the lexical category of the word, $F = \{Fname_1 = Fvalue_1, \ldots, Fname_f = Fvalue_f\}$ is a feature vector (where $Fname_i$ is the name of a feature and $Fvalue_i$ is its corresponding value), (R, L, UC, MC)+ is a list of one or more four-tuples, each representing an abstraction of a role value assignment, where R is a role variable, L is a functionality label, UC represents the relative position relation of a word and its dependent, MC is the lexical category of the modifiee for this dependency relation, and DC represents the relative ordering of the positions of a word and all of its modifiees. Notice that the SuperARV structure for *did* provides an explicit way to combine its lexical features with information concerning one consistent set of dependency links for the word that can be directly derived from its parse assignments. A SuperARV can be thought of as providing admissibility constraints on syntactic and lexical environments in which a word may be used. Once SuperARVs are assigned to a word sequence, a parse for the sentence can be produced by the constrained operation of deciding dependencies to link the SuperARVs together.

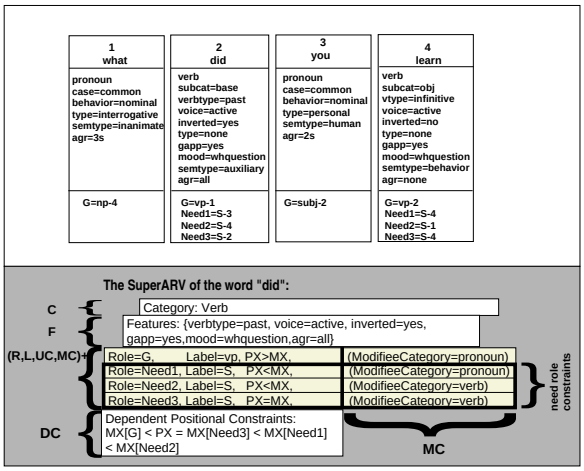

Fig. 1. The SuperARV for the word *did* given the CDG parse for the sentence *what did you learn*. Note: *G* represents the governor role; the *Need1*, *Need2*, and *Need3* roles are used to ensure that the requirements of the word are met. PX and MX represent the position of a word and its modifiee, respectively.

The model-level integration was accomplished by jointly estimating the probabilities of a sequence of words w_1^N and their SuperARV membership t_1^N:

$$Pr(w_1^N t_1^N) = \prod_{i=1}^{N} Pr(w_i t_i | w_1^{i-1} t_1^{i-1})$$

$$= \prod_{i=1}^{N} Pr(t_i | w_1^{i-1} t_1^{i-1}) \cdot Pr(w_i | w_1^{i-1} t_1^{i})$$

We use this to enable the joint prediction of words and their SuperARVs so that word identity information is tightly integrated at the model level. Note that SuperARVs serve as hidden events for constraining word prediction. Since the parametric space for the SuperARV LM is larger than a word-based LM, in [1] we have considered carefully how to interpolate lower-order n-gram probability estimations and which smoothing algorithm to use. For each smoothing algorithm investigated, we used a combination of heuristics and an entropy-based method to determine globally the lower-order n-grams to include in the interpolation, as well as their ordering. For a trigram SuperARV LM on the DARPA WSJ CSR task, the modified Kneser-Ney smoothing algorithm [7] showed the best performance. A detailed description of the best order of interpolation appears in [1].

The LM needs to be trained on CDG parses. However, since there is no CDG treebank (except for a moderate-sized corpus for the DARPA Naval Resource Management task [8] that we have annotated), we have developed a methodology to automatically transform CFG constituent bracketing into CDG annotations [1]. In addition to generating dependency structures by headword percolation [2], our transformer utilizes a rule-based method to determine lexical features and need role values for words. Although these procedures are effective, they cannot guarantee that the CDG annotations generated are completely correct. We have implemented an efficient lattice rescoring algorithm for applying the SuperARV LM with a Viterbi search using the word sequence probabilities [1].

In the next section, we describe the evaluation of the SuperARV LM on the DARPA WSJ and Hub4 (Broadcast News) CSR tasks. Evaluation on these tasks allows us to investigate our model's robustness and whether LM performance gracefully scales up to more sophisticated LVCSR tasks.

3. INVESTIGATION ACROSS TASKS

3.1. Evaluation on the DARPA WSJ CSR Task

In [1], we have compared the effectiveness of using trigram word-based, trigram POS, trigram SuperARV, and Chelba's LMs [2] in rescoring hypotheses generated by a speech recognizer for the DARPA WSJ CSR task. The training set for a word-based LM on this task is composed of the 1987-1989 LM training data containing 37,243,300 words. Our SuperARV LM must be trained on dependency parses for the transcriptions; however, the WSJ Penn Treebank covers only a small proportion of the DARPA WSJ CSR training set. For the remaining training sentences, we used the CFG constituent bracketing from the BLLIP Treebank [9], which was produced by Charniak's maximum-entropy inspired parser [10] trained on the WSJ Penn Treebank. Because the BLLIP Treebank was not proofread, it may contain incorrect parse trees. However, because it was generated using a high-quality, state-of-the-art probabilistic parser trained on the WSJ domain, we assumed that BLLIP provides parse trees with a good level of consistency and accuracy for training our LM. For testing the LMs, we used the 1992 and 1993 5K closed vocabulary and 20K open vocabulary DARPA WSJ CSR evaluation sets [1].

The trigram LM was provided by LDC while all the other LMs (i.e., the POS LM, the SuperARV LM, and Chelba's LM) were trained using information derived from the training set trees for the CSR task. Parameter tuning for the LMs on each task used the corresponding development set [1]. Each LM was then used to rescore the lattices generated by an acoustic recognizer built using HTK. Dr. Chelba also provided us with a set of lattices for 93-20K that were generated using the AT&T decoder. Table 1 shows the WER and sentence accuracy (SAC) after rescoring lattices using each LM, with the lowest WER and highest SAC for each test set presented in bold face. As can be seen from Table 1, the Super-

LM	Our HTK Lattices				Chelba's 93-20k lattices
	92-5k WER(SAC)	93-5k WER(SAC)	92-20k WER(SAC)	93-20k WER(SAC)	WER(SAC)
3gram	4.43(61.52)	6.91(43.26)	11.11(36.94)	14.74(30.52)	13.72(36.18)
POS	3.92(64.85)	6.55(47.91)	10.58(38.14)	14.52(33.22)	13.51(37.96)
SuperARV	**3.83(65.76)**	**6.24(50.23)**	**10.15(39.64)**	**14.26(36.12)**	**12.87(42.02)**
Chelba	3.85(65.45)	6.26(49.77)	10.19(39.34)	14.31(35.88)	12.93(40.48)
lattice accuracy	1.79(79.40)	2.16(73.95)	4.93(59.46)	6.65(52.11)	3.41 (68.86)

Table 1. Comparing WER and SAC (%) after rescoring lattices using each LM on the DARPA WSJ CSR 5k- and 20k- test sets. The lattice WER/SAC which defines the best accuracy possible given perfect knowledge is also provided.

ARV LM produces the best reduction in WER with Chelba's LM the second best. The SuperARV LM performs statistically significantly better than the trigram and POS LMs. Although there is no significant difference between the SuperARV LM and Chelba's LM, the former has a much lower complexity than the latter. Also, because of the additional heuristic steps required in the CFG-to-CDG conversion, our SuperARV LM was generated using a potentially greater amount of flawed training data.

3.2. Evaluation on the Hub4 Domain

A baseline word-based LM was constructed for the Hub4 task. The data sources for this model consisted of two sets of Hub4 training data (the 130 million word loosely transcribed Broadcast News corpus for LM training and the 380,000 word closely transcribed material for acoustic training) and two sets of non-Hub4 training data, that is, the North American Business News (NABN) corpus and the Switchboard-I corpus. As to the two sets of non-Hub4 training data, the former was chosen for additional coverage on business and politics, and the latter was added to increase the coverage of conversational speech characteristics (e.g., disfluencies) insufficiently covered in the Hub4 training data.

A 5-gram word-based LM was estimated for each set of Hub4 training data, and a trigram word-based LM was trained for each set of non-Hub4 training data, resulting in four LMs, each estimated by the SRILM toolkit [11] using the modified Kneser-Ney smoothing algorithm. The baseline LM was constructed by interpolating each of these four LMs. The vocabulary size of the baseline LM is 48k.

The challenge for building a SuperARV LM for the Hub4 domain is that, in contrast to the DARPA WSJ CSR task, there is no corpus of parse trees available for deriving CDG annotations for training. One way to get around this problem is to generate a set of training parse trees by using available probabilistic parsers. We investigated three of the best parsers that generate CFG bracketing: Collins' bilexical dependency parser [12], Ratnaparkhi's maximum-entropy parser [13], and Charniak's maximum-entropy inspired parser [10]. Since none of these parsers was designed to model conversational speech, we chose to generate parses for only the Hub4 LM training data and the NABN corpus, which will be denoted $\mathcal{T}$.

To construct our training Treebank for $\mathcal{T}$, we considered two important attributes of the parsers, accuracy and robustness. To evaluate parser accuracy, we consider the Labeled Precision (LP) and Labeled Recall (LR)[2] of the three parsers on the WSJ Penn Treebank [14]. As described in [10, 12, 13], these parsers have been trained, tuned, and tested on the same three data sets. Charniak's parser achieved the greatest accuracy with 89.5% LP and 89.6% LR, followed by Collins' parser at 88.3% LP and 88.1% LR, and then Ratnaparkhi's parser at 87.5% LP and 86.3% LR. To evaluate the robustness of each parser, we measured the coverage of each parser on a randomly selected subset consisting of 10% of $\mathcal{T}$, denoted $\mathcal{H}$. A sentence is *covered* by a parser if it succeeds in generating a parse tree whose yield is the entire sentence without aborting prematurely. Ratnaparkhi's parser achieved the greatest coverage at 95% of $\mathcal{H}$, followed by Collins' at 93%, and then Charniak's at 71%. To test the coverage of pairs of parsers, we generated a treebank $A \cup B$ which is comprised of all the parse trees generated by parser A for the set together with parse trees generated by parser B for any sentence in the set that failed to be covered by parser A. For $\mathcal{H}$, we generated six pairs of parsers and found that Ratnaparkhi's $\cup$ Collins' and Ratnaparkhi's $\cup$ Charniak's each cover almost 100% of $\mathcal{H}$; Collins' $\cup$ Charniak's and Collins' $\cup$ Ratnaparkhi's each cover 99%; and Charniak's $\cup$ Ratnaparkhi's and Charniak's $\cup$ Collins' each cover 98%. We concluded that, by combining the output from a pair of parsers, we would be able to obtain a better coverage of $\mathcal{T}$ than by using one alone.

Consistency of training data is important for building a good LM. Hence, we also considered the mutual consistency between the parse trees produced by the three parsers. To measure the structural consistency between the parses produced by CFG bracketing parsers and a gold standard parse, Black, Garside, and Leech [15] defined the metric *average crossing brackets* (ACB), the mean number of times per sentence that a bracketed sequence from one parser overlaps with the gold standard from the treebank such that neither is properly contained in the other. Although ACB does not account for all types of conflicting constituency, it is a practical measure for the structural consistency between two sets of parse trees. By using the output from one parser as the gold set, we can calculate the pair-wise ACB among the three parsers to measure their mutual consistency. We have calculated the ACB between parser pairs on a subset of $\mathcal{H}$ that is covered by all the three parsers (containing 14,466 sentences and about 300,000 words). The pair-wise ACB was 2.46 for Collins' and Charniak's parsers, 3.34 for Ratnaparkhi's and Collins' parsers, and 3.73 for Ratnaparkhi's and Charniak's parsers. Clearly, Collins' and Charniak's parsers have the greatest mutual structural consistency; however, their pair-wise ACB is much larger than each parser's ACB reported for the gold standard parses of the WSJ Penn Treebank, suggesting that these parsers have a greater degree of mutual inconsistency on $\mathcal{T}$.

To intelligently select parser pairs for generating a treebank for $\mathcal{T}$, we should consider the accuracy of the individual parsers, as well as the coverage and consistency of the pairs. Charniak's parser has the greatest accuracy of the three parsers, and so if ac-

[2] LP is the number of correct constituents divided by the number of constituents found by the parser and LR is the number of correct constituents divided by the number of constituents in the actual parse.

curate parsing is important for the LM it would be ideal to select it as the major contributor of parses to the treebank. Although the coverage of the parser pairs involving Ratnaparkhi's parser as the major contributor is nearly 100% on $\mathcal{H}$, that parser is also the least accurate of the three. On the other hand, the coverage for the pairs with Charniak's as the major parser is essentially equal, as is the case with pairs involving Collin's as the major parser. Furthermore, Charniak's and Collins' parsers have the greatest pairwise consistency using the ACB measure. Hence, we will evaluate treebanks that use a combination of Charniak's and Collins' parsers to parse $\mathcal{T}$, with Charniak's $\cup$ Collins', denoted *T1*, and Collins' $\cup$ Charniak's, denoted *T2*. Note that alone Charniak's parser covered only 68.4% of $\mathcal{T}$ and Collin's covered 90.5%; whereas, *T1* and *T2* cover 97.5% and 98.2% of sentences, respectively. It is important to note that *T1* and *T2* contain parse errors and that these errors should affect the quality of our CDG derivation. Hence, we will be training our SuperARV LM on training data that is probably more inconsistent and flawed than for the DARPA WSJ CSR task. However, based on the WSJ Penn Treebank parsing results, it is likely that *T1* is more accurate than *T2*.

Acoustic Cond.	Baseline LM	Interp. *T1*	Interp. *T2*
F0	12.4	10.9	11.2
F1	28.6	28.2	28.5
F2	32.1	30.7	31.8
F3	31.9	30.1	31.2
F4	22.8	22.2	22.5
F5	20.4	19.1	19.8
FX	43.4	43.2	43.3
Total	26.9	26.0	26.5

Table 2. WER (%) given acoustic conditions after N-best rescoring using the baseline word-based LM and interpolated SuperARV LMs trained on *T1* and *T2*.

A 4-gram SuperARV LM was trained using each treebank, and parameter tuning was done on a heldout data set of 20% of the training data. Since spontaneous speech characteristics (such as disfluencies) are not well represented in the training data for the SuperARV LM, we have interpolated the SuperARV LM with the baseline LM. The N-best lists (of up to 2000 hypotheses per utterance), generated by SRI's 1997 Broadcast News System for the 1996 Hub4 development test set, were rescored with a log linear combination of two types of acoustic models (crossword and non-crossword triphones), word insertion penalty, and language model. The WER was computed for the hypotheses with the highest combined scores [16].

Results are shown in Table 2. Although *T1* is not a consistent and accurate training set, the interpolated SuperARV LM results in 0.9% absolute WER reduction compared to the baseline word-based LM. Consistent with our hypothesis that a more accurate treebank (and more accurate CDG annotations) should benefit the performance of the SuperARV LM, we found that the interpolated SuperARV LM trained on *T2* obtained only 0.4% absolute WER reduction. Note that without interpolation with the baseline LM, the SuperARV LMs trained on *T1* and *T2* achieve WER of 26.1% and 26.6% on the same test set, respectively.

Table 3 demonstrates an important attribute of our SuperARV LM: the number of SuperARVs does not increase dramatically as the training data size increases from a moderate-sized Resource Management corpus to the very large Hub4 set *T1*. Hence, SuperARV LMs scale well to large data sets.

Data	RM	WSJ PTB	WSJ CSR	Hub4 *T1*
# of words	25,168	1 M	37 M	300 M
# of SuperARVs	328	538	791	1,612

Table 3. The number of SuperARVs as a function of number of words in a data set. Note that M means 1 million.

4. CONCLUSIONS AND FUTURE WORK

In cross-corpus experiments investigating the performance of the SuperARV LM, we have observed that the SuperARV structure provides a flexible framework that tightly couples a variety of knowledge sources without combinatorial explosion. Furthermore, we have found that the SuperARV LM is effective even when trained on inconsistent and flawed training data; however, developing methods to produce better quality training data are worth the effort. In future work, we will use the LM to model conversational speech phenomena. Additionally, we plan to evaluate a tighter integration between the acoustic model and SuperARV LM by applying the LM during lattice generation and expansion.

5. REFERENCES

[1] W. Wang and M. P. Harper, "The SuperARV language model: Investigating the effectiveness of tightly integrating multiple knowledge sources," in *Proceedings of Empirical Methods in Natural Language Processing*, 2002.

[2] C. Chelba, *Exploiting Syntactic Structure for Natural Language Modeling*, Ph.D. thesis, Johns Hopkins University, 2000.

[3] B. Roark, "Probabilistic top-down parsing and language modeling," *Computational Linguistics*, vol. 27, no. 2, pp. 249–276, 2001.

[4] R. Rosenfeld, "Two decades of statistical language modeling: Where do we go from here?," *Proceedings of the IEEE*, vol. 88, pp. 1270–1278, 2000.

[5] E. Charniak, "Immediate-head parsing for language models," in *Proceedings of ACL*, 2001.

[6] J. Goodman, "Probabilistic feature grammars," in *Proceedings of the Fourth International Workshop on Parsing Technologies*, 1997.

[7] S. F. Chen and J. T. Goodman, "An empirical study of smoothing techniques for language modeling," Tech. Rep., Harvard University, Computer Science Group, 1998.

[8] P. J. Price, W. Fischer, J. Bernstein, and D. Pallett, "A database for continuous speech recognition in a 1000-word domain," in *Proceedings of ICASSP*, 1988, pp. 651–654.

[9] E. Charniak, D. Blaheta, N. Ge, K. Hall, and M. Johnson, "BLLIP WSJ corpus," CD-ROM, 2000, Linguistics Data Consortium.

[10] E. Charniak, "A maximum-entropy-inspired parser," in *Proceedings of NAACL*, 2000.

[11] A. Stolcke, "SRILM–An extensible language modeling toolkit," in *Proceedings of Intl. Conf. on Spoken Language Processing*, Denver, Co, 2002, vol. 2, pp. 901–904.

[12] M. Collins, *Head-Driven Statistical Models for Natural Language Parsing*, Ph.D. thesis, University of Pennsylvania, 1999.

[13] A. Ratnaparkhi, "Learning to parse natural language with maximum entropy models," *Machine Learning*, pp. 151–176, 1999.

[14] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpus of English: The Penn Treebank," *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.

[15] E. Black, R. Garside, and G. Leech, *Statistically-driven Computer Grammars of English: The IBM/Lancaster Approach*, Rodopi, Amsterdam, 1993.

[16] A. Sankar, F. Weng, Z. Rivlin, A. Stolcke, and R. Gadde, "Development of SRI's 1997 Broadcast News transcription system," in *Proceedings of DARPA Broadcast News Transcription and Understanding Workshop*, Lansdowne, VA, 1998, pp. 91–96.