Multilingual Dependency Parsing using Global Features

Tetsuji Nakagawa
Oki Electric Industry Co., Ltd.
2−5−7 Honmachi, Chuo-ku, Osaka 541−0053, Japan
nakagawa378@oki.com

Abstract

In this paper, we describe a two-stage multilingual dependency parser used for the multilingual track of the CoNLL 2007 shared task. The system consists of two components: an unlabeled dependency parser using Gibbs sampling which can incorporate sentence-level (global) features as well as token-level (local) features, and a dependency relation labeling module based on Support Vector Machines. Experimental results show that the global features are useful in all the languages.

1 Introduction

Making use of as many informative features as possible is crucial to obtain high performance in machine learning based NLP. Recently, several methods for incorporating non-local features have been investigated, though such features often make models complex and thus complicate inference. Collins and Koo (2005) proposed a reranking method for phrase structure parsing with which any type of global features in a parse tree can be used. For dependency parsing, McDonald and Pereira (2006) proposed a method which can incorporate some types of global features, and Riedel and Clarke (2006) studied a method using integer linear programming which can incorporate global linguistic constraints. In this paper, we study dependency parsing using Gibbs sampling which can incorporate any type of global feature in a sentence. The parser determines unlabeled dependency structures only, and we attach dependency relation labels using Support Vector Machines afterwards.

We participated in the multilingual track of the CoNLL 2007 shared task (Nivre et al., 2007), and evaluated the system on data sets of 10 languages (Hajič et al., 2004; Aduriz et al., 2003; Martí et al., 2007; Chen et al., 2003; Böhmová et al., 2003; Marcus et al., 1993; Johansson and Nugues, 2007; Prokopidis et al., 2005; Csendes et al., 2005; Montemagni et al., 2003; Oflazer et al., 2003).

The rest of the paper describes the specification of the system and the evaluation results.

2 Unlabeled Dependency Parsing using Global Features

2.1 Probabilistic Model

Rosenfeld et al. (2001) proposed whole-sentence exponential language models which can incorporate arbitrary features in a sentence, and we consider here a similar probabilistic model for dependency parsing which can incorporate any sentence-level feature. Let $\mathbf{w} = w_1 \cdots w_{|\mathbf{w}|}$ denote an input sentence consisting of $|\mathbf{w}|$ tokens, and $\mathbf{h} = h_1 \cdots h_{|\mathbf{w}|}$ denote the sequence of the indices of each token's head. Root nodes of a sentence do not have heads, and we regard the index of a root node's head as zero, i.e., $h_i \in \{0, 1, \cdots, |\mathbf{w}|\} \setminus \{i\}$. We define the probability distribution of the dependency structure $\mathbf{h}$ given a sentence $\mathbf{w}$ using exponential models as follows:

$$P_{\Lambda,\mathrm{M}}(\mathbf{h}|\mathbf{w}) = \frac{1}{Z_{\Lambda,\mathrm{M}}(\mathbf{w})} Q_{\mathrm{M}}(\mathbf{h}|\mathbf{w}) \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(\mathbf{w}, \mathbf{h})\right\}, \quad (1)$$

$$Z_{\Lambda,\mathrm{M}}(\mathbf{w}) = \sum_{\mathbf{h}' \in \mathcal{H}(\mathbf{w})} Q_{\mathrm{M}}(\mathbf{h}'|\mathbf{w}) \exp\left\{\sum_{k=1}^{K} \lambda_k f_k(\mathbf{w}, \mathbf{h}')\right\}, \quad (2)$$

where $Q_{\mathrm{M}}(\mathbf{h}|\mathbf{w})$ is an initial distribution, $f_k(\mathbf{w}, \mathbf{h})$ is the k-th feature function, K is the number of feature functions, and λ_k is the weight of the k-th feature. $\mathcal{H}(\mathbf{w})$ is the set of possible configurations of heads for a given sentence $\mathbf{w}$. Although it is appropriate that $\mathcal{H}(\mathbf{w})$ is the set of projective trees for projective languages, and is the set of non-projective trees (which is a superset of the set of projective trees) for non-projective languages, in this study, we define $\mathcal{H}(\mathbf{w})$ to be the set of all the possible graphs, which contains $|\mathbf{w}|^{|\mathbf{w}|}$ elements. $P_{\Lambda,\mathrm{M}}(\mathbf{h}|\mathbf{w})$ and $Q_{\mathrm{M}}(\mathbf{h}|\mathbf{w})$ are defined over $\mathcal{H}(\mathbf{w})$[1]. The probability distribution $P_{\Lambda,\mathrm{M}}(\mathbf{h}|\mathbf{w})$ is a joint distribution of all the heads conditioned by a sentence, therefore we call this model *sentence-level model*. The feature function $f_k(\mathbf{w}, \mathbf{h})$ is defined on a sentence $\mathbf{w}$ with heads $\mathbf{h}$, and we can use any information in the sentence without the independence assumption for the heads of the tokens, therefore we call $f_k(\mathbf{w}, \mathbf{h})$

[1] $\mathcal{H}(\mathbf{w})$ is a superset of the set of non-projective trees, and is an unnecessarily large set which contains ill-formed dependency trees such as trees with cycles. This issue may cause reduction of parsing performance, but we adopt this approach for computational efficiency.

Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL 2007, pp. 952–956,
Prague, June 2007.

sentence-level (global) feature. We define initial distribution $Q_{\mathrm{M}}(\mathbf{h}|\mathbf{w})$ as the product of $q_{\mathrm{M}}(h|\mathbf{w},t)$ which is the probability distribution of the head h of each t-th token calculated with maximum entropy models:

$$Q_{\mathrm{M}}(\mathbf{h}|\mathbf{w})=\prod_{t=1}^{|\mathbf{w}|} q_{\mathrm{M}}(h_t|\mathbf{w},t), \tag{3}$$

$$q_{\mathrm{M}}(h|\mathbf{w},t)=\frac{1}{Y_{\mathrm{M}}(\mathbf{w},t)} \exp\left\{\sum_{l=1}^{L} \mu_l g_l(\mathbf{w},t,h)\right\}, \tag{4}$$

$$Y_{\mathrm{M}}(\mathbf{w},t)=\sum_{\substack{h'=0\\h'\neq t}}^{|\mathbf{w}|} \exp\left\{\sum_{l=1}^{L} \mu_l g_l(\mathbf{w},t,h')\right\}, \tag{5}$$

where $g_l(\mathbf{w},t,h)$ is the l-th feature function, L is the number of feature functions, and μ_l is the weight of the l-th feature. $q_{\mathrm{M}}(h|\mathbf{w},t)$ is a model of the head of a single token, calculated independently from other tokens, therefore we call $q_{\mathrm{M}}(h|\mathbf{w},t)$ *token-level model*, and $g_l(\mathbf{w},t,h)$ *token-level (local) feature*.

2.2 Decoding and Parameter Estimation

Let us consider how to find the optimal solution $\hat{\mathbf{h}}$, given a sentence $\mathbf{w}$, parameters of the sentence-level model $\Lambda = \{\lambda_1,\cdots,\lambda_K\}$, and parameters of the token-level model $\mathrm{M} = \{\mu_1,\cdots,\mu_L\}$. Since the probabilistic model contains global features and efficient algorithms such as dynamic programming cannot be used, we use Gibbs sampling to obtain an approximated solution. Gibbs sampling can efficiently generate samples from high-dimensional probability distributions with complex dependencies among variables (Andrieu et al., 2003), and we assume that R samples $\{\mathbf{h}^{(1)},\cdots,\mathbf{h}^{(R)}\}$ are generated from $P_{\Lambda,\mathrm{M}}(\mathbf{h}|\mathbf{w})$ using Gibbs sampling. Then, the marginal distribution of the head of the t-th token given $\mathbf{w}$, $P_t(h|\mathbf{w})$, is approximately calculated as follows:

$$\begin{aligned}P_t(h|\mathbf{w}) &= \sum_{\substack{h_1,\cdots,h_{t-1},h_{t+1},\cdots,h_{|\mathbf{w}|}\\h_t=h}} P_{\Lambda,\mathrm{M}}(\mathbf{h}|\mathbf{w}),\\ &=\sum_{\mathbf{h}} P_{\Lambda,\mathrm{M}}(\mathbf{h}|\mathbf{w})\delta(h,h_t) \simeq \frac{1}{R}\sum_{r=1}^{R}\delta(h,h_t^{(r)}), \end{aligned}\tag{6}$$

where $\delta(i,j)$ is the Kronecker delta. In order to find a solution using the marginal distribution, we adopt the maximum spanning tree (MST) framework proposed by McDonald et al. (2005a). In this framework, scores for possible edges in dependency graphs are defined, and the optimal dependency tree is found as the MST in which the summation of the edge scores is maximized. Let $s(i,j)$ denote the score of the edge from a parent node (head) i to a child node (dependent) j. We define $s(i,j)$ as follows:

$$s(i,j)=\log P_j(i|\mathbf{w}). \tag{7}$$

We use the logarithm of the marginal distribution because the summation of edge scores is maximized by the MST search algorithms but the product of the marginal distributions should be maximized. The best projective parse tree is obtained using the Eisner algorithm (Eisner, 1996) with the scores, and the best non-projective one is obtained using the Chu-Liu-Edmonds (CLE) algorithm (McDonald et al., 2005b).

Although in this method, the factored score $s(i,j)$ is used to measure likelihood of dependency trees, the score is calculated taking a whole sentence into consideration using Gibbs sampling.

Next, we explain how to estimate the parameters of our models, given training data consisting of N examples $\{\langle\mathbf{w}^1,\mathbf{h}^1\rangle,\cdots,\langle\mathbf{w}^N,\mathbf{h}^N\rangle\}$. In order to estimate the parameters of the token-level model $\mathrm{M} = \{\mu_1,\cdots,\mu_L\}$, we use maximum a posteriori estimation with Gaussian priors. We define the following objective function $\mathcal{M}$:

$$\mathcal{M}=\log\prod_{n=1}^{N} Q_{\mathrm{M}}(\mathbf{h}^n|\mathbf{w}^n) - \frac{1}{2\sigma^2}\sum_{l=1}^{L}\mu_l^2, \tag{8}$$

where σ is a hyper parameter of Gaussian priors. The optimal parameters M which maximize $\mathcal{M}$ can be obtained by quasi-Newton methods such as the L-BFGS algorithm with above $\mathcal{M}$ and its partial derivatives. The parameters of the sentence-level model $\Lambda = \{\lambda_1,\cdots,\lambda_K\}$ can also be estimated in a similar way with the following objective function $\mathcal{L}$ after the parameters of the token-level model are estimated.

$$\mathcal{L}=\log\prod_{n=1}^{N} P_{\Lambda,\mathrm{M}}(\mathbf{h}^n|\mathbf{w}^n) - \frac{1}{2\sigma'^2}\sum_{k=1}^{K}\lambda_k^2. \tag{9}$$

This objective function and its partial derivative contain summations over all the possible configurations which are difficult to calculate. We approximately calculate these values using static Monte Carlo (not MCMC) methods with fixed S samples $\{\mathbf{h}^{n(1)},\cdots,\mathbf{h}^{n(S)}\}$ generated from $Q_{\mathrm{M}}(\mathbf{h}|\mathbf{w}^n)$[2]:

$$\log Z_{\Lambda,\mathrm{M}}(\mathbf{w}^n)\simeq\log\frac{1}{S}\sum_{s=1}^{S}\exp\left\{\sum_{k=1}^{K}\lambda_k f_k(\mathbf{w}^n,\mathbf{h}^{n(s)})\right\}, \tag{10}$$

$$\begin{aligned}&\sum_{\mathbf{h}'\in\mathcal{H}(\mathbf{w}^n)} P_{\Lambda,\mathrm{M}}(\mathbf{h}'|\mathbf{w}^n) f_k(\mathbf{w}^n,\mathbf{h}')\\ &\simeq\frac{1}{S}\sum_{s=1}^{S}\frac{f_k(\mathbf{w}^n,\mathbf{h}^{n(s)})}{Z_{\Lambda,\mathrm{M}}(\mathbf{w}^n)}\exp\left\{\sum_{k'=1}^{K}\lambda_{k'} f_{k'}(\mathbf{w}^n,\mathbf{h}^{n(s)})\right\}. \end{aligned}\tag{11}$$

[2] Static Monte Carlo methods become inefficient when the dimension of the probabilistic distribution is high, and more sophisticated methods would be used for accurate parameter estimation.

2.3 Local Features

The token-level features used in the system are the same as those used in MSTParser version 0.4.2[3]. The features include lexical forms and (coarse and fine) POS tags of parent tokens, child tokens, their surrounding tokens, and tokens between the child and the parent. The direction and the distance from a parent to its child, and the FEATS fields of the parent and the child which are split into elements and then combined are also included. Features that appeared less than 5 times in training data are ignored.

2.4 Global Features

Global features can capture any information in dependency trees, and the following nine types of global features are used (In the following, *parent node* means a head token, and *child node* means a dependent token):

Child Unigram+Parent+Grandparent This feature template is a 4-tuple consisting of (1) a child node, (2) its parent node, (3) the direction from the parent node to the child node, and (4) the grandparent node.

Each node in the feature template is expanded to its lexical form and coarse POS tag in order to obtain actual features. Features that appeared in four or less sentences are ignored. The same procedure is applied to the following other features.

Child Bigram+Parent This feature template is a 4-tuple consisting of (1) a child node, (2) its parent node, (3) the direction from the parent node to the child node, and (4) the nearest outer sibling node (the nearest sibling node which exists on the opposite side of the parent node) of the child node. This feature template is almost the same as the one used by McDonald and Pereira (2006).

Child Bigram+Parent+Grandparent This feature template is a 5-tuple. The first four elements (1)–(4) are the same as the *Child Bigram+Parent* feature template, and the additional element (5) is the grandparent node.

Child Trigram+Parent This feature template is a 5-tuple. The first four elements (1)–(4) are the same as the *Child Bigram+Parent* feature template, and the additional element (5) is the next nearest outer sibling node of the child node.

[3]http://sourceforge.net/projects/mstparser

Parent+All Children This feature template is a tuple with more than one element. The first element is a parent node, and the other elements are all of its child nodes.

Parent+All Children+Grandparent This feature template is a tuple with more than two elements. The elements other than the last one are the same as the *Parent+All Children* feature template, and the last element is the grandparent node.

Child+Ancestor This feature template is a 2-tuple consisting of (1) a child node, and (2) one of its ancestor nodes.

Acyclic This feature type has one of two values, *true* if the dependency tree is acyclic, or *false* otherwise.

Projective This feature type has one of two values, *true* if the dependency tree is projective, or *false* otherwise.

3 Dependency Relation Labeling

3.1 Model

Dependency relation labeling can be handled as a multi-class classification problem, and we use Support Vector Machines (SVMs) which have been successfully applied to many NLP tasks. Solving large-scale multi-class classification problem with SVMs requires substantial computational resources, so we use the revision learning method (Nakagawa et al., 2002). The revision learning method combines a probabilistic model which has smaller computational cost with a binary classifier which has higher generalization capacity. In the method, the latter classifier revises the output of the former model to conduct multi-class classification with higher accuracy and reasonable computational cost. In this study, we use maximum entropy (ME) models as the probabilistic model and SVMs with the second order polynomial kernel as the binary classifier. The dependency label of each node is determined independently of the labeling of other nodes.

3.2 Features

As the features for SVMs to predict the dependency relation label of the i-th token, we use the lexical forms, coarse and fine POS tags, and FEATS fields of the i-th and the h_i-th tokens. We also use lexical forms and POS tags of the tokens surrounding and in between them (i.e. the j-th token where $j \in \{j | \min\{i, h_i\} - 1 \leq j \leq \max\{i, h_i\} + 1\}$), the grandparent ($h_{h_i}$-th) token, the sibling tokens of i (the j'-th token where $j' \in \{j' | h_{j'} = h_i, j' \neq i\}$),

	Arabic	Basque	Catalan	Chinese	Czech	English	Greek	Hungarian	Italian	Turkish	*Average*
LAS	75.08	72.56	87.90	83.84	80.19	88.41	76.31	76.74	83.61	78.22	80.29
UAS	86.09	81.04	92.86	88.88	86.28	90.13	84.08	82.49	87.91	85.77	86.55

Table 1: Results of Multilingual Dependency Parsing

Algorithm	*Features*	Arabic	Basque	Catalan	Chinese	Czech	English	Greek	Hungarian	Italian	Turkish
Eisner	local	85.15	80.20	91.75	86.75	84.19	88.65	83.31	80.27	86.72	84.82
(proj.)	+global	**86.09**	81.00	**92.86**	**88.88**	85.99	**90.13**	**84.08**	81.55	87.91	84.82
CLE	local	84.80	80.39	91.23	86.71	84.21	88.07	83.03	81.15	86.85	85.35
(non-proj.)	+global	85.83	**81.04**	92.64	88.84	**86.28**	90.05	83.87	**82.49**	**87.97**	**85.77**

Table 2: Unlabeled Attachment Scores in Different Settings (underlined values indicate submitted results, and bold values indicate the highest scores)

and the child tokens of i (the j''-th token where $j'' \in \{j''|h_{j''} = i\})^4$. As the features for ME models, a subset of them is used since ME models are used just for reducing the search space, and do not need so many features.

4 Results and Analysis

In order to tune the system, we split each training data set into two parts, and used the first half for training and the remaining half for testing in development. The CLE algorithm was used for Basque, Czech, Hungarian and Turkish, and the Eisner algorithm was used for the others. We used lemmas for Catalan, Czech, Greek and Italian, and word forms for all others. The values of the parameters to be fixed were chosen as $R = 500$, $S = 200$, $\sigma = 0.25$, and $\sigma' = 0.25$. With these parameter settings, training took 247 hours, and testing took 343 minutes on an Opteron 250 processor.

Table 1 shows the evaluation results on the test sets. Accuracy was measured with the labeled attachment score (LAS) and the unlabeled attachment score (UAS). Among the participating systems in the shared task, we obtained the second best average accuracy in the labeled attachment score, and the best average accuracy in the unlabeled attachment score. Compared with other systems, the gap between our labeled and unlabeled scores is relatively big. In this study, labeling of dependency relations was performed in a separate post-processing step, and each label was predicted independently. The labeled scores may be improved if the parsing process and the labeling process are performed at the same time, and dependencies among labels are taken into account.

We conducted experiments with different settings. Table 2 shows the results measured with the unlabeled attachment score. In the table, **Eisner** and **CLE** indicate that the Eisner algorithm and the CLE algorithm are used in decoding, and **local** and **+global** indicate that local features alone, and local and global features together are used. The CLE algorithm performed better than the Eisner algorithm for Basque, Czech, Hungarian, Italian and Turkish. All of these data sets except Italian contain relatively a large number of non-projective sentences (the percentage of sentences with at least one non-projective relation in the training data is over 20% (Nivre et al., 2007)), though the Greek data set, on which the Eisner algorithm performed better, also contains many non-projective sentences (20.3%).

By using the global features, the accuracy was improved in all the cases except for Turkish with the Eisner algorithm (Table 2). The increase was rather large in Chinese and Czech. When the global features were used in these languages, the dependency accuracy for tokens whose heads had conjunctions as parts-of-speech was notably improved; from 80.5% to 86.0% in Chinese (Eisner), and from 73.2% to 77.6% in Czech (CLE). We investigated the trained global models, and found that *Parent+All Children* features, whose parents were conjunctions and whose children had compatible classes, had large positive weights, and those whose children had incompatible classes had large negative weights. A feature with a larger weight is generally more influential. Riedel and Clarke (2006) suggested to use linguistic constraints such as “arguments of a coordination must have compatible word classes,” and such constraint seemed to be represented by the features in our models.

5 Conclusion

In this study, we applied a dependency parser using global features to multilingual dependency parsing. Evaluation results showed that the use of global features was effective to obtain higher accuracy in multilingual dependency parsing. Improving dependency relation labeling is left for future work.

[4]Although polynomial kernels of SVMs can implicitly handle combined features, some of combined features were also included explicitly because using unnecessarily high order polynomial kernels decreases performance.

References

A. Abeillé, editor. 2003. *Treebanks: Building and Using Parsed Corpora*. Kluwer.

I. Aduriz, M. J. Aranzabe, J. M. Arriola, A. Atutxa, A. Diaz de Ilarraza, A. Garmendia, and M. Oronoz. 2003. Construction of a Basque Dependency Treebank. In *Proc. of the 2nd Workshop on Treebanks and Linguistic Theories (TLT)*, pages 201–204.

C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. 2003. An Introduction to MCMC for Machine Learning. *Machine Learning*, 50:5–43.

A. Böhmová, J. Hajič, E. Hajičová, and B. Hladká. 2003. The PDT: a 3-level annotation scenario. In Abeillé (Abeillé, 2003), chapter 7, pages 103–127.

K. Chen, C. Luo, M. Chang, F. Chen, C. Chen, C. Huang, and Z. Gao. 2003. Sinica Treebank: Design Criteria, Representational Issues and Implementation. In Abeillé (Abeillé, 2003), chapter 13, pages 231–248.

M. Collins and T. Koo. 2005. Discriminative Reranking for Natural Language Parsing. *Computational Linguistics*, 31(1):25–69.

D. Csendes, J. Csirik, T. Gyimóthy, and A. Kocsor. 2005. *The Szeged Treebank*. Springer.

J. Eisner. 1996. Three New Probabilistic Models for Dependency Parsing: An Exploration. In *Proc. of COLING '96*, pages 340–345.

J. Hajič, O. Smrž, P. Zemánek, J. Šnaidauf, and E. Beška. 2004. Prague Arabic Dependency Treebank: Development in Data and Tools. In *Proc. of the NEMLAR Intern. Conf. on Arabic Language Resources and Tools*, pages 110–117.

R. Johansson and P. Nugues. 2007. Extended Constituent-to-Dependency Conversion for English. In *Proc. of the 16th Nordic Conference on Computational Linguistics (NODALIDA)*.

M. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

M. A. Martí, M. Taulé, L. Màrquez, and M. Bertran. 2007. CESS-ECE: A Multilingual and Multilevel Annotated Corpus. Available for download from: http://www.lsi.upc.edu/∼mbertran/cess-ece/.

R. McDonald and F. Pereira. 2006. Online Learning of Approximate Dependency Parsing Algorithms. In *Proc. of EACL 2006*, pages 81–88.

R. McDonald, K. Crammer, and F. Pereira. 2005a. Online Large-Margin Training of Dependency Parsers. In *Proc. of ACL 2005*, pages 91–98.

R. McDonald, F. Pereira, K. Ribarow, and J. Hajic. 2005b. Non-projective dependency parsing using Spanning Tree Algorithms. In *Proc. of HLT/EMNLP 2005*, pages 523–530.

S. Montemagni, F. Barsotti, M. Battista, N. Calzolari, O. Corazzari, A. Lenci, A. Zampolli, F. Fanciulli, M. Massetani, R. Raffaelli, R. Basili, M. T. Pazienza, D. Saracino, F. Zanzotto, N. Nana, F. Pianesi, and R. Delmonte. 2003. Building the Italian Syntactic-Semantic Treebank. In Abeillé (Abeillé, 2003), chapter 11, pages 189–210.

T. Nakagawa, T. Kudo, and Y. Matsumoto. 2002. Revision Learning and its Application to Part-of-speech Tagging. In *Proc. of ACL 2002*, pages 497–504.

J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proc. of the CoNLL 2007 Shared Task. Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

K. Oflazer, B. Say, D. Zeynep Hakkani-Tür, and G. Tür. 2003. Building a Turkish Treebank. In Abeillé (Abeillé, 2003), chapter 15, pages 261–277.

P. Prokopidis, E. Desypri, M. Koutsombogera, H. Papageorgiou, and S. Piperidis. 2005. Theoretical and Practical Issues in the Construction of a Greek Dependency Treebank. In *Proc. of the 4th Workshop on Treebanks and Linguistic Theories (TLT)*, pages 149–160.

S. Riedel and J. Clarke. 2006. Incremental Integer Linear Programming for Non-projective Dependency Parsing. In *Proc. of EMNLP 2006*, pages 129–137.

R. Rosenfeld, S. F. Chen, and X. Zhu. 2001. Whole-Sentence Exponential Language Models: A Vehicle For Linguistic-Statistical Integration. *Computers Speech and Language*, 15(1):55–73.