Transduction Recursive Auto-Associative Memory: Learning Bilingual Compositional Distributed Vector Representations of Inversion Transduction Grammars

Karteek Addanki **Dekai Wu**
HKUST
Human Language Technology Center
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
{vskaddanki|dekai}@cs.ust.hk

Abstract

We introduce TRAAM, or Transduction RAAM, a fully bilingual generalization of Pollack's (1990) monolingual Recursive Auto-Associative Memory neural network model, in which each distributed vector represents a bilingual constituent—i.e., an instance of a transduction rule, which specifies a relation between two monolingual constituents and how their subconstituents should be permuted. Bilingual terminals are special cases of bilingual constituents, where a vector represents either (1) a bilingual token —a token-to-token or "word-to-word" translation rule —or (2) a bilingual segment—a segment-to-segment or "phrase-to-phrase" translation rule. TRAAMs have properties that appear attractive for bilingual grammar induction and statistical machine translation applications. Training of TRAAM drives both the autoencoder weights and the vector representations to evolve, such that similar bilingual constituents tend to have more similar vectors.

1 Introduction

We introduce Transduction RAAM—or TRAAM for short—a recurrent neural network model that generalizes the monolingual RAAM model of Pollack (1990) to a distributed vector representation of compositionally structured transduction grammars (Aho and Ullman, 1972) that is fully bilingual from top to bottom. In RAAM, which stands for Recursive Auto-Associative Memory, using feature vectors to characterize constituents at every level of a parse tree has the advantages that (1) the entire context of all subtrees inside the constituent can be efficiently captured in the feature vectors, (2) the learned representations generalize well because similar feature vectors represent similar constituents or segments, and (3) representations can be automatically learned so as to maximize prediction accuracy for various tasks using semi-supervised learning. We argue that different, but analogous, properties are desirable for bilingual structured translation models.

Unlike RAAM, where each distributed vector represents a monolingual token or constituent, each distributed vector in TRAAM represents a *bilingual* constituent or **biconstituent**—that is, an instance of a transduction rule, which asserts a relation between two monolingual constituents, as well as specifying how to permute their subconstituents in translation. Bilingual terminals, or **biterminals**, are special cases of biconstituents where a vector represents either (1) a **bitoken**—a token-to-token or "word-to-word" translation rule —or (2) a **bisegment**—a segment-to-segment or "phrase-to-phrase" translation rule.

The properties of TRAAMs are attractive for machine translation applications. As with RAAM, TRAAMs can be trained via backpropagation training, which simultaneously evolves both the autoencoder weights and the biconstituent vector representations. As with RAAM, the evolution of the vector representations within the hidden layer performs automatic feature induction, and for many applications can obviate the need for manual feature engineering. However, the result is that similar vectors tend to represent similar *biconstituents*, rather than monolingual constituents.

The learned vector representations thus tend to form clusters of similar *translation relations* instead of merely similar strings. That is, TRAAM clusters represent soft nonterminal categories of cross-lingual relations and translation patterns, as opposed to soft nonterminal categories of monolingual strings as in RAAM.

Also, TRAAMs inherently make full simultaneous use of both input *and* output language fea-

Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, pages 112–121,
October 25, 2014, Doha, Qatar.

tures, recursively, in an elegant integrated fashion. TRAAM does not make restrictive *a priori* assumptions of conditional independence between input and output language features. When evolving the biconstituent vector representations, generalization occurs over similar input *and* output structural characteristics simultaneously. In most recurrent neural network applications to machine translation to date, only input side features or only output language features are used. Even in the few previous cases where recurrent neural networks have employed both input and output language features for machine translation, the models have typically been factored so that their recursive portion is applied only to either the input or output language, but not both.

As with RAAM, the objective criteria for training can be adjusted to reflect accuracy on numerous different kinds of tasks, biasing the direction that vector representations evolve toward. But again, TRAAM's learned vector representations support making predictions that simultaneously make use of both input and output structural characteristics. For example, TRAAM has the ability to take into account the structure of both input and output subtree characteristics while making predictions on reordering them. Similarly, for specific cross-lingual tasks such as word alignment, sense disambiguation, or machine translation, classifiers can simultaneously be trained in conjunction with evolving the vector representations to optimize task-specific accuracy (Chrisman, 1991).

In this paper we use as examples binary biparse trees consistent with transduction grammars in a 2-normal form, which by definition are inversion transduction grammars (Wu, 1997) since they are binary rank. This is not a requirement for TRAAM, which in general can be formed for transduction grammars of any rank. Moreover, with distributed vector representations, the notion of nonterminal categories in TRAAM is that of soft membership, unlike in symbolically represented transduction grammars. We start with bracketed training data that contains no bilingual category labels (like training data for Bracketing ITGs or BITGs). Training results in self-organizing clusters that have been automatically induced, representing soft nonterminal categories (unlike BITGs, which do not have differentiated nonterminal categories).

2 Related work

TRAAM builds on different aspects of a spectrum of previous work. A large body of work exists on various different types of self-organizing recurrent neural network approaches to modeling recursive structure, but mostly in monolingual modeling. Even in applications to machine translation or cross-lingual modeling, the typical practice has been to insert neural network scoring components while still maintaining older SMT modeling assumptions like bags-of-words/phrases, "shake'n'bake" translation that relies heavily on strong monolingual language models, and log-linear models —in contrast to TRAAM's fully integrated bilingual approach. Here we survey representative work across the spectrum.

2.1 Monolingual related work

Distributed vector representations have long been used for n-gram language modeling; these continuous-valued models exploit the generalization capabilities of neural networks, although there is no hidden contextual or hierarchical structure as in RAAM. Schwenk (2010) applies one such language model within an SMT system.

In the simple recurrent neural networks (RNNs or SRNs) of Elman (1990), hidden layer representations are fed back to the input to dynamically represent an aggregate of the immediate contextual history. More recently, the probabilistic NNLMs of Bengio *et al.* (2003) and Bengio *et al.* (2009) follow in this vein.

To represent hierarchical tree structure using vector representations, one simple family of approaches employs convolutional networks, as in Lee *et al.* (2009) for example. Collobert and Weston (2008) use a convolution neural network layer quite effectively to learn vector representations for words which are then used in a host of NLP tasks such as POS tagging, chunking, and semantic role labeling.

RAAM approaches, and related recursive autoencoder approaches, can be more flexible than convolutional networks. Like SRNs, they can be extended in numerous ways. The URAAM (Unification RAAM) model of Stolcke and Wu (1992) extended RAAM to demonstrate the possibility of using neural networks to perform more sophisticated operations like unification directly upon the distributed vector representations of hierarchical

feature structures. Socher *et al.* (2011) used monolingual recursive autoencoders for sentiment prediction, with or without parse tree information; this was perhaps the first use of a RAAM style approach on a large scale NLP task, albeit monolingual. Scheible and Schütze (2013) automatically simplified the monolingual tree structures generated by recursive autoencoders, validated the simplified structures via manual evaluation, and showed that sentiment classification accuracy is not affected.

2.2 Bilingual related work

The majority of work on learning bilingual distributed vector representations has not made use of recursive approaches or hidden contextual or compositional structure, as in the bilingual word embedding learning of Klementiev *et al.* (2012) or the bilingual phrase embedding learning of Gao *et al.* (2014). Schwenk (2012) uses a non-recursive neural network to predict phrase translation probabilities in conventional phrase-based SMT.

Attempts have been made to generalize the distributed vector representations of monolingual *n*-gram language models, avoiding any hidden contextual or hierarchical structure. Working within the framework of *n*-gram translation models, Son *et al.* (2012) generalize left-to-right monolingual *n*-gram models to bilingual *n*-grams, and study bilingual variants of class-based *n*-grams. However, their model does not allow tackling the challenge of modeling cross-lingual constituent order, as TRAAM does; instead it relies on the assumption that some other preprocessor has already managed to accurately re-order the words of the input sentence into exactly the order of words in the output sentence.

Similarly, generalizations of monolingual SRNs to the bilingual case have been studied. Zou *et al.* (2013) generalize the monolingual recurrent NNLM model of Bengio *et al.* (2009) to learn bilingual word embeddings using conventional SMT word alignments, and demonstrate that the resulting embeddings outperform the baselines in word semantic similarity. They also add a single semantic similarity feature induced with bilingual embeddings to a phrase-based SMT log-linear model, and report improvements in BLEU. Compared to TRAAM, however, they only learn non-compositional features, with distributed vectors only representing biterminals (as opposed to bi-constituents or bilingual subtrees), and so other mechanisms for combining biterminal scores still need to be used to handle hierarchical structure, as opposed to seamlessly being integrated into the distributed vector representation model. Devlin *et al.* (2014) obtain translation accuracy improvements by extending the probabilistic NNLMs of Bengio *et al.* (2003), which are used for the output language, by adding input language context features. Unlike TRAAM, neither of these approaches symmetrically models the recursive structure of both the input and output language sides.

For convolutional network approaches, Kalchbrenner and Blunsom (2013) use a recurrent probabilistic model to generate a representation of the source sentence and then generate the target sentence from this representation. This use of input language context to bias translation choices is in some sense a neural network analogy to the PSD (phrase sense disambiguation) approach for context-dependent translation probabilities of Carpuat and Wu (2007). Unlike TRAAM, the model does not contain structural constraints, and permutation of phrases must still be done in conventional PBSMT "shake'n'bake" style by relying mostly on a language model (in their case, a NNLM).

A few applications of monolingual RAAM-style recursive autoencoders to bilingual tasks have also appeared. For cross-lingual document classification, Hermann and Blunsom (2014) use two separate monolingual fixed vector composition networks, one for each language. One provides the training signal for the other, and training is only on the embeddings.

Li *et al.* (2013) described a use of monolingual recursive autoencoders within maximum entropy ITGs. They replace their earlier model for predicting reordering based on the first and the last tokens in a constituent, by instead using the context vector generated using the recursive autoencoder. Only input language context is used, unlike TRAAM which can use the input and output language contexts equally.

Autoencoders have also been applied to SMT in a very different way by Zhao *et al.* (2014) but without recursion and not for learning distributed vector representations of words; rather, they used non-recursive autoencoders to compress very high-dimensional bilingual sparse features down to low-dimensional feature vectors, so that MIRA or PRO

could be used to optimize the log-linear model weights.

3 Representing transduction grammars with TRAAM

As a recurrent neural network representation of a transduction grammar, TRAAM learns bilingual distributed representations that parallel the structural composition of a transduction grammar. As with transduction grammars, the learned representations are symmetric and model structured relational correlations between the input and output languages. The induced feature vectors in effect represent soft categories of cross-lingual relations and translations. The TRAAM model integrates elegantly with the transduction grammar formalism and aims to model the compositional structure of the transduction grammar as opposed to incorporating external alignment information. It is straightforward to formulate TRAAMs for arbitrary syntax directed transduction grammars; here we shall describe an example of a TRAAM model for an inversion transduction grammar (ITG).

Formally, an ITG is a tuple $\langle N, \Sigma, \Delta, R, S \rangle$, where N is a finite nonempty set of nonterminal symbols, Σ is a finite set of terminal symbols in L_0, Δ is a finite set of terminal symbols in L_1, R is a finite nonempty set of inversion transduction rules and $S \in N$ is a designated start symbol. A normal-form ITG consists of rules in one of the following four forms:

$$S \rightarrow A,\ A \rightarrow [BC]\,,\ A \rightarrow \langle BC \rangle,\ A \rightarrow e/f$$

where $S \in N$ is the start symbol, $A, B, C \in N$ are nonterminal symbols and e/f is a biterminal. A biterminal is a pair of symbol strings: $\Sigma^* \times \Delta^*$, where at least one of the strings have to be nonempty. The square and angled brackets signal straight and inverted order respectively. With straight order, both the L_0 and the L_1 productions are generated left-to-right, but with inverted order, the L_1 production is generated right-to-left.

In the distributed TRAAM representation of the ITG, we represent each bispan, using a feature vector v of dimension d that represents a fuzzy encoding of all the nonterminals that could generate it. This is in contrast to the ITG model where each nonterminal that generates a bispan has to be enumerated separately. Feature vectors corresponding to larger bispans are compositionally generated from smaller bispans using a *compressor* network which takes two feature vectors of dimension d, corresponding to the smaller bispans and generates the feature vector of dimension d corresponding to the larger bispan. A single bit corresponding to straight or inverted order is also fed as an input to the compressor network. The compressor network in TRAAM serves a similar role as the syntactic rules in the symbolic ITG, but keeps the encoding fuzzy. Figure 2 shows the straight and inverted syntactic rules and the corresponding inputs to the compressor network. Modeling of unary rules (with start symbol on the left hand side) although similar, is beyond the scope of this paper.

It is easy to demonstrate that TRAAM models are capable of representing any symbolic ITG model. All the nonterminals representing a bispan can be encoded as a bit vector in the feature vector of the bispan. Using the universal approximation theorem of neural networks (Hornik *et al.*, 1989), an encoder with a single hidden layer can represent any set of syntactic rules. Similarly, all TRAAM models can be represented using a symbolic ITG by assuming a unique nonterminal label for every feature vector. Therefore, TRAAM and ITGs represent two equivalent classes of models for representing compositional bilingual relations.

It is important to note that although both TRAAM and ITG models might be equivalent, the fuzzy encoding of nonterminals in TRAAM is suitable for modeling the generalizations in bilingual relations without exploding the search space unlike the symbolic models. This property of TRAAM makes it attractive for bilingual category learning and machine translation applications as long as appropriate language bias and objective functions are determined.

Given our objective of inducing categories of bilingual relations in an unsupervised manner, we bias our TRAAM model by using a simple nonlinear activation function to be our compressor, similar to the monolingual recursive autoencoder model proposed by Socher *et al.* (2011). Having a single layer in our compressor provides the necessary *language bias* by forcing the network to capture the generalizations while reducing the dimensions of the input vectors. We use tanh as the nonlinear activation function and the compressor accepts two vectors c_1 and c_2 of dimension d corresponding to the nonterminals of the smaller bispans and a single bit o corresponding to the in-

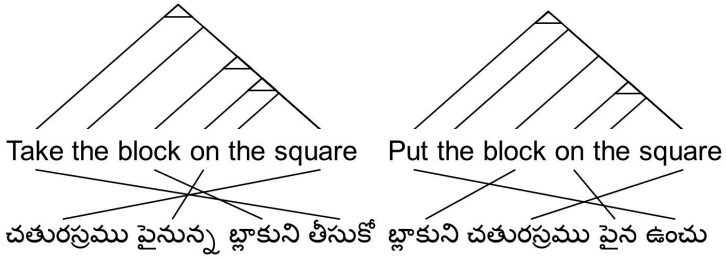

Figure 1: Example of English-Telugu biparse trees where inversion depends on output language sense.

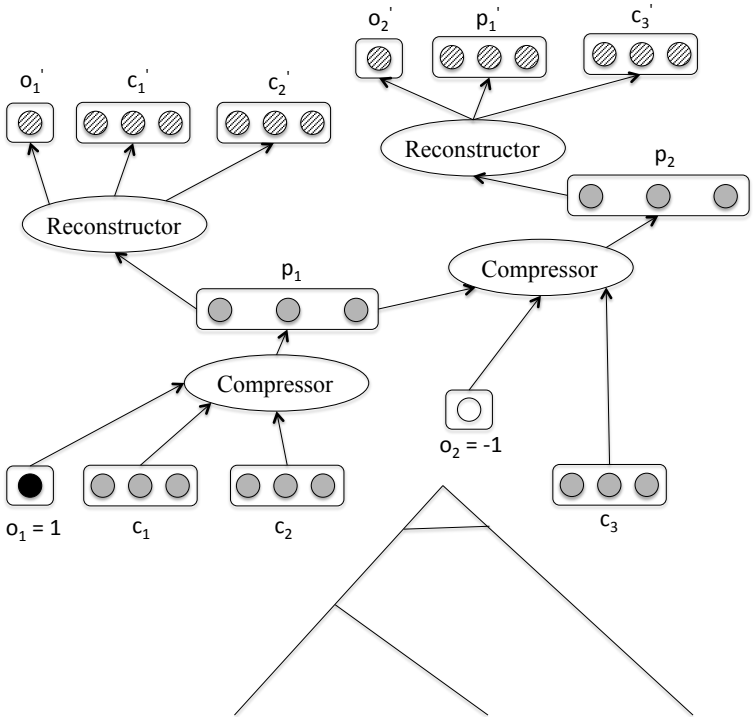

Figure 2: Architecture of TRAAM.

version order and generates a vector p of dimension d corresponding to the larger bispan generated by combining the two smaller bispans as shown in Figure 2. The vector p then serves as the input for the successive combinations of the larger bispan with other bispans.

$$p = \tanh(W_1[o; c_1; c_2] + b_1) \quad (1)$$

where W_1 and b_1 are the weight matrix and the bias vector of the encoder network.

To ensure that the computed vector p captures the fuzzy encodings of its children and the inversion order, we use a *reconstructor* network which attempts to reconstruct the inversion order and the feature vectors corresponding of its children. We use the error in reconstruction as our objective function and train our model to minimize the reconstruction error over all the nodes in the biparse tree. The reconstructor network in our TRAAM model can be replaced by any other network that enables the computed feature vector representations to be optimized for the given task. In our current implementation, we reconstruct the inversion order o' and the child vectors c_1' and c_2' using another nonlinear activation function as follows:

$$[o'; c_1'; c_2'] = \tanh(W_2 p + b_2) \quad (2)$$

where W_2 and b_2 are the weight matrix and the bias vector of the reconstructor network.

4 Bilingual training

4.1 Initialization

The weights and the biases of the compressor and the reconstructor networks of the TRAAM model are randomly initialized. *Bisegment embeddings*

corresponding to the leaf nodes (biterminals in the symbolic ITG notation) in the biparse trees are also initialized randomly. These constitute the model parameters and are optimized to minimize our objective function of reconstruction error. The parse trees for providing the structural constraints are generated by a bracketing inversion transduction grammar (BITG) induced in a purely unsupervised fashion, according to the algorithm in Saers *et al.* (2009). Due to constraints on the training time, we consider only the Viterbi biparse trees according to the BITG instead of all the biparse trees in the forest.

4.2 Computing feature vectors

We compute the feature vectors at each internal node in the biparse tree, similar to the feedforward pass in a neural network. We topologically sort all the nodes in the biparse tree and set the feature vector of each node in the topologically sorted order as follows:

- If the node is a leaf node, the feature vector is the corresponding bisegment embedding.
- Else, the *biconstituent embedding* corresponding to the internal node is generated using the feature vectors of the children and the inversion order using Equation 1. We also normalize the length of the computed feature vector so as to prevent the network from making the biconstituent embedding arbitrarily small in magnitude (Socher *et al.*, 2011).

4.3 Feature vector optimization

We train our current implementation of TRAAM, by optimizing the model parameters to minimize an objective function based on the reconstruction error over all the nodes in the biparse trees. The objective function is defined as a linear combination of the l2 norm of the reconstruction error of the children and the cross-entropy loss of reconstructing the inversion order. We define the error at each internal node n as follows:

$$E_n = \frac{\alpha}{2}|[c_1;c_2] - [c_1';c_2']|^2 - (1-\alpha)$$
$$[(1-o)\log(1-o') + (1+o)\log(1+o')]$$

where c_1, c_2, o correspond to the left child, right child and inversion order, c_1', c_2', o' are the respective reconstructions and α is the linear weighting factor. The global objective function J is the sum of the error function at all internal nodes n in the biparse trees averaged over the total number of sentences T in the corpus. A regularization parameter λ is used on the norm of the model parameters θ to avoid overfitting.

$$J = \frac{1}{T}\Sigma_n E_n + \lambda||\theta||^2 \quad (3)$$

As the bisegment embeddings are also a part of the model parameters, the optimization objective is similar to a moving target training objective Rohwer (1990). We use backpropagation with structure Goller and Kuchler (1996) to compute the gradients efficiently. L-BFGS algorithm Liu and Nocedal (1989) is used in order to minimize the loss function.

5 Bilingual representation learning

We expect the TRAAM model to generate clusters over cross-lingual relations similar to RAAM models on monolingual data. We test this hypothesis by bilingually training our model using a parallel English-Telugu blocks world dataset. The dataset is kept simple to better understand the nature of clusters. Our dataset comprises of commands which involves manipulating different colored objects over different shapes.

5.1 Example

Figure 1 shows the biparse trees for two English-Telugu sentence pairs. The preposition on in English translates to పైనున్న(pinunna) and పైన(pina) respectively in the first and second sentence pairs because in the first sentence block is described by its position on the square, whereas in the second sentence block is the subject and square is the object. Since Telugu is a language with an SOV structure, the verbs ఉంచు(vunchu) and తీసుకో(teesuko) occur at the end for both sentences.

The sentences in 1 illustrate the importance of modeling bilingual relations simultaneously instead of focusing only on the input or output language as the cross-lingual structural relations are sensitive to both the input and output language context. For example, the constituent whose input side is block on the square, the corresponding output language tree structure is determined by whether or not on is translated to పైనున్న (pinunna) or పైన(pina).

In symbolic frameworks such as ITGs, such relations are encoded using different nonterminal categories. However, inducing such cate-

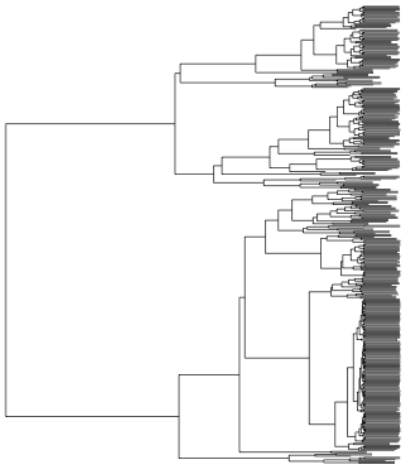

Figure 3: Clustering of biconstituents in the Telugu-English data.

gories within a symbolic framework in an unsupervised manner creates extremely challenging combinatorial scaling issues. TRAAM models are a promising approach for tackling this problem, since the vector representations learned using the TRAAM model inherently yield soft syntactic category membership properties, despite being trained only with the unlabeled structural constraints of simple BITG-style data.

5.2 Biconstituent clustering

The soft membership properties of learned distributed vector representations can be explored via cluster analysis. To illustrate, we trained a TRAAM network bilingually using the algorithm in Section 4, and obtained feature vector representations for each unique biconstituent. Clustering the obtained feature vectors reveals emergence of fuzzy nonterminal categories, as shown in Figure 3. It is important to note that each point in the vector space corresponds to a tree-structured biconstituent as opposed to merely a flat bilingual phrase, as same surface forms with different tree structures will have different vectors.

As the full cluster tree is too unwieldy, Figure 4 zooms in to shows an enlarged version of a portion of the clustering, along with the corresponding bracketed bilingual structures. One can observe that the cluster represents the biconstituents that describe the object by its position on another object. We can deduce this from the fact that only a single sense of on/పైనున్న(pinnuna) seems to be occurring in *all* the biconstituents of the cluster. Manual inspection of other clusters reveals such similarities despite noise expected to be introduced by the sparsity of our dataset.

6 Conclusion

We have introduced a fully bilingual generalization of Pollack's (1990) monolingual Recursive Auto-Associative Memory neural network model, TRAAM, in which each distributed vector represents a bilingual constituent—i.e., an instance of a transduction rule, which specifies a relation between two monolingual constituents and how their subconstituents should be permuted. Bilingual terminals are special cases of bilingual constituents, where a vector represents either (1) a bilingual token—a token-to-token or "word-to-word" translation rule—or (2) a bilingual segment—a segment-to-segment or "phrase-to-phrase" translation rule.

TRAAMs can be used for arbitrary rank SDTGs (syntax-directed transduction grammars, a.k.a. synchronous context-free grammars). Although our discussions in this paper have focused on biparse trees from SDTGs in a 2-normal form, which by definition are ITGs due to the binary rank, nothing prevents TRAAMs from being applied to higher-rank transduction grammars.

We believe TRAAMs are worth detailed exploration as their intrinsic properties address key problems in bilingual grammar induction and sta-

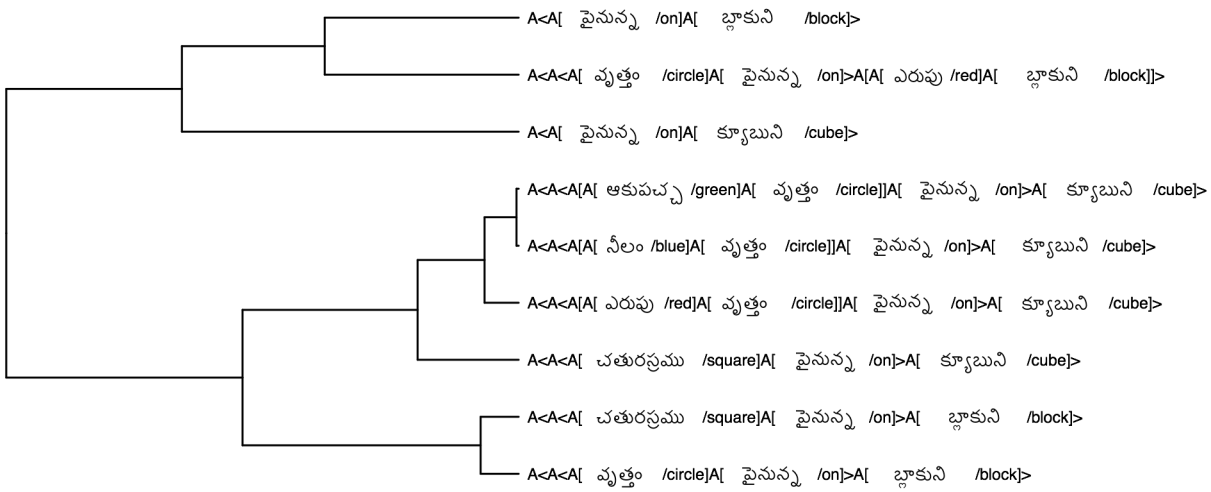

Figure 4: Typical zoomed view into the Telugu-English biconstituent clusters from Figure 3.

tistical machine translation—their sensitivity to both input and output language context means that the learned vector representations tend to reflect the similarity of *bilingual* rather than monolingual constituents, which is what is needed to induce differentiated bilingual nonterminal categories.

7 Acknowledgments

This material is based upon work supported in part by the Defense Advanced Research Projects Agency (DARPA) under BOLT contract nos. HR0011-12-C-0014 and HR0011-12-C-0016, and GALE contract nos. HR0011-06-C-0022 and HR0011-06-C-0023; by the European Union under the FP7 grant agreement no. 287658; and by the Hong Kong Research Grants Council (RGC) research grants GRF620811, GRF621008, and GRF612806. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA, the EU, or RGC.

References

Alfred V. Aho and Jeffrey D. Ullman. *The Theory of Parsing, Translation, and Compiling*. Prentice-Halll, Englewood Cliffs, New Jersey, 1972.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

Marine Carpuat and Dekai Wu. Context-dependent phrasal translation lexicons for statistical machine translation. In *11th Machine Translation Summit (MT Summit XI)*, pages 73–80, 2007.

Lonnie Chrisman. Learning recursive distributed representations for holistic computation. *Connection Science*, 3(4):345–366, 1991.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA, 2008. ACM.

Jacob Devlin, Rabih Zbib, Zhongqiang Huang, Thomas Lamar, Richard Schwartz, and John Makhoul. Fast and robust neural network joint models for statistical machine translation. In

52nd Annual Meeting of the Association for Computational Linguistics, 2014.

Jeffrey L Elman. Finding structure in time. *Cognitive science*, 14(2):179–211, 1990.

Jianfeng Gao, Xiaodong He, Wen-tau Yih, and Li Deng. Learning continuous phrase representations for translation modeling. In *52nd Annual Meeting of the Association for Computational Linguistics (Short Papers)*, 2014.

Christoph Goller and Andreas Kuchler. Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1, pages 347–352. IEEE, 1996.

Karl Moritz Hermann and Phil Blunsom. Multilingual models for compositional distributed semantics. In *52nd Annual Meeting of the Association for Computational Linguistics*, volume abs/1404.4641, 2014.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *EMNLP*, pages 1700–1709, 2013.

Alexandre Klementiev, Ivan Titov, and Binod Bhattarai. Inducing crosslingual distributed representations of words. In *24th International Conference on Computational Linguistics (COLING 2012)*. Citeseer, 2012.

Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616. ACM, 2009.

Peng Li, Yang Liu, and Maosong Sun. Recursive autoencoders for itg-based translation. In *EMNLP*, pages 567–577, 2013.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.

Jordan B Pollack. Recursive distributed representations. *Artificial Intelligence*, 46(1):77–105, 1990.

Richard Rohwer. The "moving targets" training algorithm. In *Neural Networks*, pages 100–109. Springer, 1990.

Markus Saers, Joakim Nivre, and Dekai Wu. Learning stochastic bracketing inversion transduction grammars with a cubic time biparsing algorithm. In *11th International Conference on Parsing Technologies (IWPT'09)*, pages 29–32, Paris, France, October 2009.

Christian Scheible and Hinrich Schütze. Cutting recursive autoencoder trees. In *1st International Conference on Learning Representations (ICLR 2013)*, Scottsdale, Arizona, May 2013.

Holger Schwenk. Continuous-space language models for statistical machine translation. In *The Prague Bulletin of Mathematical Linguistics*, volume 93, pages 137–146, 2010.

Holger Schwenk. Continuous space translation models for phrase-based statistical machine translation. In *Proceedings of COLING 2012: Posters*, pages 1071—1080. Citeseer, 2012.

Richard Socher, Jeffrey Pennington, Eric H Huang, Andrew Y Ng, and Christopher D Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 151–161. Association for Computational Linguistics, 2011.

Le Hai Son, Alexandre Allauzen, and François Yvon. Continuous space translation models with neural networks. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 39–48. Association for Computational Linguistics, 2012.

Andreas Stolcke and Dekai Wu. Tree matching with recursive distributed representations. In *AAAI 1992 Workshop on Integrating Neural and Symbolic Processes —The Cognitive Dimension*, 1992.

Dekai Wu. Stochastic inversion transduction grammars and bilingual parsing of parallel corpora. *Computational Linguistics*, 23(3):377–403, 1997.

Bing Zhao, Yik-Cheung Tam, and Jing Zheng. An autoencoder with bilingual sparse features for improved statistical machine translation. In

IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP), 2014.

Will Y Zou, Richard Socher, Daniel M Cer, and Christopher D Manning. Bilingual word embeddings for phrase-based machine translation. In *EMNLP*, pages 1393–1398, 2013.