MT/IE: Cross-lingual Open Information Extraction with Neural Sequence-to-Sequence Models

Sheng Zhang and **Kevin Duh** and **Benjamin Van Durme**
Johns Hopkins University
{zsheng2, kevinduh, vandurme}@cs.jhu.edu

Abstract

Cross-lingual information extraction is the task of distilling facts from foreign language (e.g. Chinese text) into representations in another language that is preferred by the user (e.g. English tuples). Conventional pipeline solutions decompose the task as machine translation followed by information extraction (or vice versa). We propose a joint solution with a neural sequence model, and show that it outperforms the pipeline in a cross-lingual open information extraction setting by 1-4 BLEU and 0.5-0.8 F_1.

1 Introduction

Suppose an English-speaking user is faced with the daunting task of distilling facts from a collection of Chinese documents. One solution is to first translate the Chinese documents into English using a Machine Translation (MT) service, then extract the facts using an English-based Information Extraction (IE) engine. Unfortunately, imperfect translations negatively impact the IE engine, which may have been trained to expect natural English input (Sudo et al., 2004). Another approach is to first run a Chinese-based IE engine and then translate the results, but this relies on IE resources in the source language. Such problems with pipeline systems compound when the IE engine relies on parsers or other analytics as features.

We propose to solve the cross-lingual IE task with a joint approach. Further, we focus on *Open* IE, which allows for an open set of semantic relations between a predicate and its arguments. Open IE in the monolingual setting has shown to be useful in a wide range of tasks, such as question answering (Fader et al., 2014), ontology learning (Suchanek, 2014), and summarization (Christensen et al., 2013). A variety of work has achieved compelling results at monolingual Open IE (Banko et al., 2007; Fader et al., 2011; Angeli et al., 2015). But we are not aware of efforts that focus on both the cross-lingual and open aspects of cross-lingual Open IE, despite significant work in related areas, such as cross-lingual IE on a closed, pre-defined set of events/entities (Sudo et al., 2004; Parton et al., 2009; Ji, 2009; Snover et al., 2011; Ji et al., 2016), or bootstrapping of monolingual Open IE systems in multiple languages (Faruqui and Kumar, 2015; Kozhevnikov and Titov, 2013; van der Plas et al., 2014).

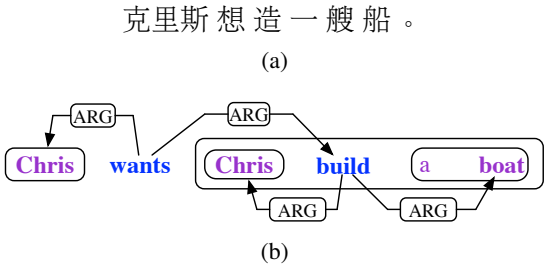

Figure 1: Example of input (a) and output (b) of cross-lingual Open IE.

Inspired by the recent success of neural models in machine translation (Kalchbrenner and Blunsom, 2013; Cho et al., 2014; Bahdanau et al., 2014), syntactic parsing (Vinyals et al., 2015; Choe and Charniak, 2016), and semantic parsing (Dong and Lapata, 2016), we propose a sequence-to-sequence model that enables end-to-end cross-lingual Open IE. Essentially, we recast the problem as structured translation: the model encodes natural-language sentences and decodes predicate-argument forms (Figure 1). We show that the joint approach outperforms the pipeline on various metrics, and that the neural model is critical for the joint approach because of its capability in generating complex open IE patterns.

2 Cross-lingual Open IE Framework

Open IE involves the extraction of relations whose schema need not be specified in advance; typically the relation name is represented by the text linking the arguments, which can be identified by manually-written patterns and/or parse trees. We define our extractions based on PredPatt[1] (White et al., 2016), a lightweight tool for identifying predicate-argument structures with a set of Universal Dependencies (UD) based patterns.

PredPatt represents predicates and arguments in a tree structure where a special dependency `ARG` is built between a predicate head token and its arguments' head tokens, and original UD dependencies within predicate phrases and argument phrases are kept. For example, Fig 1b shows a tree structure identified by PredPatt from the sentence: "*Chris wants to build a boat.*"

Our framework assumes the availability of a bitext, e.g. a corpus of Chinese sentences and their English translations. We run PredPatt on the target side (e.g. English) to obtain (Chinese sentence, English PredPatt) pairs. This is used to train a cross-lingual Open IE system that maps directly from Chinese sentence to English PredPatt representations. Besides the UD parser required for running PredPatt on the target side, our framework requires no additional resources.

Compared to existing Open IE (Banko et al., 2007; Fader et al., 2011; Angeli et al., 2015), the use of manual patterns on Universal Dependencies means that the rules are interpretable, extensible and language-agnostic, which makes PredPatt a linguistically well-founded component for cross-lingual Open IE. Note that our joint model is agnostic to the IE representation, and can be adapted to other Open IE frameworks.

3 Proposed Method

Our goal is to learn a model which directly maps a sentence input A in the source language into predicate-argument structures output B in the target language. Formally, we regard the input as a sequence $A = x_1, \cdots, x_{|A|}$, and use a *linearized* representation of the predicate-argument structure as the output sequence $B = y_1, \cdots, y_{|B|}$. While tree-based decoders are conceivable (Zhang et al., 2016), linearization of structured outputs to sequences simplifies decoding and has been shown effective in, e.g. (Vinyals et al., 2015), especially when a model with strong memory capabilities (e.g. LSTM's) are employed. Our model maps A into B using a conditional probability which is decomposed as:

$$P(B \mid A) = \prod_{t=1}^{|B|} P(y_t \mid y_1, \cdots, y_{t-1}, A) \quad (1)$$

3.1 Linearized PredPatt Representations

We begin by defining a linear form for our PredPatt predicate-argument structures. To convert a tree structure such as Figure 1b to a linear sequence, we first take an in-order traversal of every node (token). We then label each token with the type it belongs to: p for a predicate token, a for an argument token, p_h for a predicate head token, and a_h for an argument head token. We insert parentheses to either the beginning or the end of an argument, and we insert brackets to either the beginning or the end of a predicate. Fig 2 shows the linearized PredPatt for the sentence: "*Chris wants to build a boat.*".

[(Chris:a_h) wants:p_h [(Chris:a_h) build:p_h (a:a boat:a_h)]]

Figure 2: Linearized PredPatt Output

To recover the predicate-argument tree structure, we simply build it recursively from the outermost brackets. At each layer of the tree, parentheses help recover argument nodes. The labels a_h and p_h help identify the head token of a predicate and an argument, respectively. We define that an auto-generated linearized PredPatt is malformed if it has unmatched brackets or parentheses, or a predicate (or an argument) has zero or more than one head token.

3.2 Seq2Seq Model

Our sequence-to-sequence (Seq2Seq) model consists of an encoder which encodes a sentence input A into a vector representation, and a decoder which learns to decode a sequence of linearized PredPatt output B conditioned on encoded vector.

We adopt a model similar to that which is used in neural machine translation (Bahdanau et al., 2014). The encoder uses an L-layer bidirectional RNN (Schuster and Paliwal, 1997) which consists of a forward RNN reading inputs from x_1 to $x_{|A|}$ and a backward RNN reading inputs in reverse from $x_{|A|}$ to x_1. Let $\overrightarrow{\boldsymbol{h}_i^l} \in \mathbb{R}^n$ denote

[1] https://github.com/hltcoe/PredPatt

the forward hidden state at time step i and layer l; it is computed by states at the previous time-step and at a lower layer: $\overrightarrow{\boldsymbol{h}}_i^l = \overrightarrow{f}(\overrightarrow{\boldsymbol{h}_{i-1}^l}, \overrightarrow{\boldsymbol{h}_i^{l-1}})$ where $\overrightarrow{f}$ is a nonlinear LSTM unit (Hochreiter and Schmidhuber, 1997). The lowest layer $\overrightarrow{\boldsymbol{h}}_i^0$ is the word embedding of the token x_i. The backward hidden state $\overleftarrow{\boldsymbol{h}}_i^l$ is computed similarly using another LSTM, and the representation of each token x_i is the concatenation of the top-layers: $\boldsymbol{h}_t = [\overrightarrow{\boldsymbol{h}_i^L}^\intercal, \overleftarrow{\boldsymbol{h}_i^L}^\intercal]^\intercal$.

The decoder is an L-layer RNN which predicts the next token y_i, given all the previous words $\boldsymbol{y}_{<i} = y_1, \cdots, y_{i-1}$ and the context vector $\boldsymbol{c}_i$ that captures the attention to the encoder side (Bahdanau et al., 2014; Luong et al., 2015), computed as a weighted sum of hidden representations: $\boldsymbol{c}_i = \sum_{j=1}^{l} a_{ij}\boldsymbol{h}_j$. The weight a_{ij} is computed by

$$
\begin{aligned}
a_{ij} &= \frac{\exp(e_{ij})}{\sum_{k=1}^{l}\exp(e_{ik})} \\
e_{ij} &= v_a^\intercal \tanh(\sum_{l=1}^{L} \boldsymbol{W}_a^l s_{i-1}^l + \boldsymbol{U}_a \boldsymbol{h}_j)
\end{aligned}
\tag{2}
$$

where $v_a \in \mathbb{R}^n$, $\boldsymbol{W}_a^l \in \mathbb{R}^{n\times n}$ and $\boldsymbol{U}_a \in \mathbb{R}^{n\times 2n}$ are weight matrices.

The conditional probability of the next token y_i is defined as:

$$
\begin{aligned}
P(y_i \mid \boldsymbol{y}_{<i}, A) &= g(y_i, \boldsymbol{s}_i^L, \boldsymbol{c}_i) \\
&= \text{softmax}(\boldsymbol{U}_o \boldsymbol{s}_i^L + \boldsymbol{C}_o \boldsymbol{c}_i)[y_i]
\end{aligned}
$$

where $\boldsymbol{U}_o \in \mathbb{R}^{|V_B|\times n}$ and $\boldsymbol{C}_o \in \mathbb{R}^{|V_B|\times 2n}$ are weight matrices.$[j]$ indexes jth element of a vector. s_i^L is the top-layer hidden state at time step i, computed recursively by $\boldsymbol{s}_i^l = f(\boldsymbol{s}_{i-1}^l, \boldsymbol{s}_i^{l-1}, \boldsymbol{c}_i)$ where $\boldsymbol{s}_i^0 = \boldsymbol{W}_B[y_{i-1}]$ is the word vector of the previous token y_{i-1}, with $\boldsymbol{W}_B \in \mathbb{R}^{|V_B|\times n}$ being a parameter matrix.

Training: The objective function is to minimize the negative log likelihood of the target linearized PredPatt given the sentence input:

$$
\text{minimize} - \sum_{(A,B)\in\mathcal{D}} \sum_{i}^{|A|} \log P(y_i \mid \boldsymbol{y}_{<i}, A) \tag{3}
$$

where $\mathcal{D}$ is the batch of training pairs, and $P(y_i \mid \boldsymbol{y}_{<i}, A)$ is computed by Eq.(3).

Inference: We use greedy search to decode tokens one by one: $\hat{y}_i = \arg\max_{y_i \in V_B} P(y_i | \hat{\boldsymbol{y}}_{<i}, A)$

4 Experiments

We describe the data for evaluation, hyperparameters, comparing approaches and evaluation results.[2]

Data: We choose Chinese as the source language and English as the target language. To prepare the data for evaluation, we first collect about 2M Chinese-English parallel sentences[3]. We then tokenize Chinese sentences using Stanford Word Segmenter (Chang et al., 2008), and generate English linearized PredPatt by running SyntaxNet Parser (Andor et al., 2016) and PredPatt (White et al., 2016) on English sentences. After removing long sequences (length>50), we result in 990K pairs of Chinese sentences and English linearized PredPatt, which are then randomly divided for training (950K), validation (10K) and test (40K). Fig 3 shows the statistics of the data. Note that in general, the linearized PredPatt sequences are not short, and can contain multiple predicates.

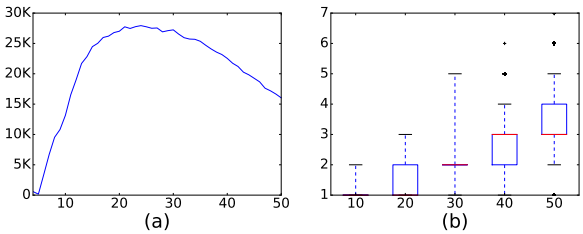

Figure 3: Data Statistics: (a) Number of data pairs with respect to the lengths of English linearized PredPatt; (b) Boxplot of numbers of English predicate with respect to the lengths of English linearized PredPatt.

Hyperparameters: Our proposed model (**Joint-Seq2Seq**) is trained using the Adam optimiser (Kingma and Ba, 2014), with mini-batch size 64 and step size 200. Both encoder and decoder have 2 layers and hidden state size 512, but different LSTM parameters sampled from $\mathcal{U}$(-0.05,0.05). Vocabulary size is 40K for both sides. Dropout (rate=0.5) is applied to non-recurrent connections (Srivastava et al., 2014). Gradients are clipped when their norm is bigger than 5 (Pascanu et al., 2013). We use sampled softmax to speed up training (Jean et al., 2015).

Comparisons: As an alternative, we train a phrase-based machine translation system,

[2]The code is available at `https://github.com/sheng-z/cross-lingual-open-ie`.

[3]The data comes from the GALE project; the largest bitexts are LDC2007E103 and LDC2006G05

Moses (Koehn et al., 2007), directly on the same data we used to train **Joint-Seq2Seq**, i.e. pairs of Chinese sentences and English linearized PredPatt. We call this system **Joint-Moses**. We also train a **Pipeline** system which consists of a Moses system that translates Chinese sentence to English *sentence*, followed by SyntaxNet Parser (Andor et al., 2016) for Universal Dependency parsing on English, and PredPatt for predicate-argument identification.

Results: We regard the generation of linearized PredPatt or linearized predicates[4] as a translation problem, and use BLEU score (Papineni et al., 2002) for evaluation. As shown in Table 1, Joint Seq2Seq achieves the best BLEU scores, with an improvement 1.7 BLEU for linearized PredPatt and improvement of 4.3 BLEU for linearized predicates compared to Pipeline.

	PredPatt	Predicates
Pipeline	17.19	17.24
Joint Moses	18.34	16.43
Joint Seq2Seq	**18.94**	**21.55**

Table 1: Evaluation results (BLEU) of linearized PredPatt and linearized predicates.

We also evaluate predicates in the same vein as event detection evaluation using the weighted F_1 score.[5] There are totally 9,535 predicate tokens in the test data. To enable a coarser-grain evaluation, we also partitioned these predicates into k clusters ($k \in \{150, 1252\}$) and evaluated F_1 on the cluster identities.The clusters are obtained by running Bisecting k-Means algorithm on pre-trained word embeddings (Rastogi et al., 2015).[6] Table 2 shows the F_1 scores: Joint Seq2Seq outperforms Pipeline by 0.5-0.8 at different granularities.

An important aspect of the auto-generated linearized PredPatt is its recoverability. Table 3 shows the number of unrecoverable outputs (including empty or malformed ones). Since the last step in Pipeline is to run PredPatt, Pipeline generates no malformed output. However, 15% of its outputs are empty. In contrast, Joint Seq2Seq generates no empty output and very few malformed ones (1%). Joint Moses also generates no empty output, but a large amount (84%) of its outputs is malformed.

	k=150	k=1252	k=9535
Pipeline	32.95	28.73	27.20
Joint Moses	32.56	27.94	25.43
Joint Seq2Seq	**33.67**	**29.21**	**28.03**

Table 2: Evaluation results (weighted F_1) of predicates at different cluster granularities.

Pipeline	Joint Moses	Joint Seq2Seq
5965(15%)	33178(84%)	557(1%)

Table 3: Number of unrecoverable outputs.

Table 4 shows an example output. While some arguments (e.g., "*The **focus** of focus*" in Table 4) are not correct, the output of Joint Seq2Seq is closest to the gold in terms of translation. Pipeline has the higher precision in predicting the same predicate head tokens as the gold, but its overall meaning is less close. Joint Moses often generates unrecoverable outputs (e.g., the predicate in Table 4 has two head tokens: "***focus***" and "***related***".)

zh_sent:	重点 审计 关注 与 老百姓 生活 密切 相关 的 专项 资金 .
en_sent:	The focus of the auditing will be on special item funds that are closely related to people 's living .
gold:	[(The **focus** of the auditing) will be on special special **funds** [(special item **funds**) are closely **related** to (people 's **living**)]]
Pipeline:	[(the key auditing **concern** and ordinary people) are closely **related** to (the **life** of the special funds)]
Joint-Moses:	[(the auditing **focus** (**attention**) to (**life**) with (ordinary **people**) are closely **related** to (the special funds)]
Joint-Seq2Seq:	[(The **focus** of focus) **focused** on (the special **collection** of the specific funds) [(the special **funds**) **related** to (people 's **lives**)]]

Table 4: Example output. Arguments are shown in blue, and predicates shown in purple. Head tokens are underlined in bold. Token labels are omitted.

5 Conclusions

We focus on the problem of cross-lingual open IE, and propose a joint solution based on a neu-

[4]In linearized predicates, arguments are replaced by placeholders. For example, the linearized PredPatt in Fig 2 becomes "[?arg wants:p_h Sth:= [?arg build:p_h ?arg]]" after replacement.

[5]Weighted F_1 is the weighted average of individual F_1 for each predicate, with weights proportional to predicate frequencies in the test data. We use token-level F_1 score (Liu et al., 2015) which gives partial credits to partial matches.

[6]Downloaded from: `https://github.com/se4u/mvlsa`.

ral sequence-to-sequence model. Our joint approach outperforms the pipeline solution by 1-4 BLEU and 0.5-0.8 F_1. Future work includes minimum risk training (Shen et al., 2016) for directly optimizing the cross-lingual open IE metrics of interest. Furthermore, as PredPatt works on any language that has UD parsers available, we plan to evaluate cross-lingual Open IE on other target languages. We are also interested in exploring how our cross-lingual open IE output, which contains rich information about predicates and arguments, can be used to facilitate existing IE tasks like relation extraction, event detection, and named entity recognition in a cross-lingual setting.

Acknowledgments

This work was supported in part by the JHU Human Language Technology Center of Excellence (HLTCOE), and DARPA LORELEI. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes. The views and conclusions contained in this publication are those of the authors and should not be interpreted as representing official policies or endorsements of DARPA or the U.S. Government.

References

Daniel Andor, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov, and Michael Collins. 2016. Globally normalized transition-based neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2442–2452, Berlin, Germany, August. Association for Computational Linguistics.

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, Beijing, China, July. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 2670–2676, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Pi-Chuan Chang, Michel Galley, and Christopher D. Manning. 2008. Optimizing Chinese word segmentation for machine translation performance. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 224–232, Columbus, Ohio, June. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October. Association for Computational Linguistics.

Do Kook Choe and Eugene Charniak. 2016. Parsing as language modeling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2331–2336, Austin, Texas, November. Association for Computational Linguistics.

Janara Christensen, Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1173, Atlanta, Georgia, June. Association for Computational Linguistics.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 33–43, Berlin, Germany, August. Association for Computational Linguistics.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open Question Answering Over Curated and Extracted Knowledge Bases. In *KDD*.

Manaal Faruqui and Shankar Kumar. 2015. Multilingual open relation extraction using cross-lingual projection. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1351–1356, Denver, Colorado, May–June. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1–10, Beijing, China, July. Association for Computational Linguistics.

Heng Ji, Joel Nothman, and Hoa Trang Dang. 2016. Overview of tac-kbp2016 tri-lingual edl and its impact on end-to-end kbp. In *Proceedings of the Text Analysis Conference (TAC)*.

Heng Ji. 2009. Cross-lingual predicate cluster acquisition to improve bilingual event extraction by inductive learning. In *Proceedings of the Workshop on Unsupervised and Minimally Supervised Learning of Lexical Semantics*, pages 27–35, Boulder, Colorado, USA, June. Association for Computational Linguistics.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA, October. Association for Computational Linguistics.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics.

Mikhail Kozhevnikov and Ivan Titov. 2013. Cross-lingual transfer of semantic role labeling models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1190–1200, Sofia, Bulgaria, August. Association for Computational Linguistics.

Zhengzhong Liu, Teruko Mitamura, and Eduard Hovy. 2015. Evaluation algorithms for event nugget detection: A pilot study. In *Proceedings of the 3rd Workshop on EVENTS at the NAACL-HLT*, pages 53–57.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Kristen Parton, Kathleen R. McKeown, Bob Coyne, Mona T. Diab, Ralph Grishman, Dilek Hakkani-Tür, Mary Harper, Heng Ji, Wei Yun Ma, Adam Meyers, Sara Stolbach, Ang Sun, Gokhan Tur, Wei Xu, and Sibel Yaman. 2009. Who, what, when, where, why? comparing multiple approaches to the cross-lingual 5w task. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 423–431, Suntec, Singapore, August. Association for Computational Linguistics.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of The 30th International Conference on Machine Learning*, pages 1310–1318.

Pushpendre Rastogi, Benjamin Van Durme, and Raman Arora. 2015. Multiview lsa: Representation learning via generalized cca. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 556–566, Denver, Colorado, May–June. Association for Computational Linguistics.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2016. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1692, Berlin, Germany, August. Association for Computational Linguistics.

Matthew Snover, Xiang Li, Wen-Pin Lin, Zheng Chen, Suzanne Tamang, Mingmin Ge, Adam Lee, Qi Li, Hao Li, Sam Anzaroot, and Heng Ji. 2011. Cross-lingual slot filling from comparable corpora. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, BUCC '11, pages 110–119, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Fabian Suchanek. 2014. Information extraction for ontology learning. *Lehmann and Völker [2 6]*, pages 135–151.

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. 2004. Cross-lingual information extraction system evaluation. In *Proceedings of the 20th international Conference on Computational Linguistics*, page 882. Association for Computational Linguistics.

Lonneke van der Plas, Marianna Apidianaki, and Chenhua Chen. 2014. Global methods for cross-lingual semantic role and predicate labelling. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1279–1290, Dublin, Ireland, August. Dublin City University and Association for Computational Linguistics.

Oriol Vinyals, Łukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*, pages 2773–2781.

Aaron Steven White, Drew Reisinger, Keisuke Sakaguchi, Tim Vieira, Sheng Zhang, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2016. Universal decompositional semantics on universal dependencies. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1713–1723, Austin, Texas, November. Association for Computational Linguistics.

Xingxing Zhang, Liang Lu, and Mirella Lapata. 2016. Top-down tree long short-term memory networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 310–320, San Diego, California, June. Association for Computational Linguistics.