Context Models for OOV Word Translation in Low-Resource Languages

Angli Liu anglil@cs.washington.edu
Department of Computer Science, University of Washington, Seattle, WA, 98195

Katrin Kirchhoff kk2@u.washington.edu
Department of Electrical Engineering, University of Washington, Seattle, WA, 98195

Abstract

Out-of-vocabulary word translation is a major problem for the translation of low-resource languages that suffer from a lack of parallel training data. This paper evaluates the contributions of target-language context models towards the translation of OOV words, specifically in those cases where OOV translations are derived from external knowledge sources, such as dictionaries. We develop both neural and non-neural context models and evaluate them within both phrase-based and self-attention based neural machine translation systems. Our results show that neural language models that integrate additional context beyond the current sentence are the most effective in disambiguating possible OOV word translations. We present an efficient second-pass lattice-rescoring method for wide-context neural language models and demonstrate performance improvements over state-of-the-art self-attention based neural MT systems in five out of six low-resource language pairs.

1 Introduction

Translation of out-of-vocabulary (OOV) words (words occurring in the test data but not in the training data) is of major importance in statistical machine translation (MT). It is a particularly difficult problem in low-resource languages, i.e., languages for which parallel training data is extremely sparse, requiring recourse to techniques that are complementary to standard statistical machine translation approaches. The approaches described in this paper were developed for scenarios where the training data comprises at most 100k sentences pairs. Most previous studies in this area have focused on how to generate translation candidates for OOV words, either by segmentation into subword units, projection from other languages, or by leveraging external knowledge sources like dictionaries. Often, however, these methods generate multiple candidates for each OOV word, and the MT system is insufficiently trained to choose the appropriate translation according to the context.

In this paper we address this problem by utilizing more sophisticated context models based on target-language information. In particular, we develop wide-context models that incorporate information from context beyond current sentence boundaries to resolve translation ambiguity.

We compare these against models incorporating information from the current sentence only, and evaluate neural models vs. count-based sentence completion and graph reranking models. All are evaluated within both phrase-based and attention-based neural machine translation models for 6 low-resource language pairs. Our paper makes several contributions:

- We evaluate recently proposed neural machine translation (NMT) architectures (purely attention-based neural MT) on low-resource languages and show that, contrary to previous results obtained with sequence-to-sequence models, neural MT performs similarly to phrase-based machine translation (PBMT) in these scenarios, without modifications to the basic model.
- We develop and compare several wide-context target-language based models for translation disambiguation and find that document-level neural language models are the most effective at resolving translation ambiguities.
- We present an efficient lattice rescoring algorithm for wide-context neural language models.
- We compare our approach against directly adding external translation resources to the training data and show that our approach provides small but consistent improvements on five out of six language pairs.

The rest of the paper is organized as follows. Section 2 discusses prior work on OOV translation. Section 3 describes our general approach and presents various context models for translation disambiguation. Section 4 describes the datasets and baseline systems. Section 5 provides experimental results followed by a final conclusion in Section 6.

2 Prior Work

Several techniques for OOV word translation have been developed in the past. The simplest of these involves copying OOV words from the source sentence to the MT output. This is a reasonable procedure if most OOV words can be assumed to be numbers or named entities that do not require transliteration. In traditional PBMT systems, the unknown words can simply be passed through to the output. NMT models typically map all OOVs (as well as rare words) to a single unknown word token. Luong et al. (2014) trained an NMT system using external word alignment information, which allowed the system to output positional information about OOVs, which were then translated using a dictionary trained from parallel data. Working within the context of neural sequence-to-sequence models with attention, Bahdanau et al. (2014) and Jean et al. (2014) pursued the same strategy, with the exception that alignment information was derived from the attention layer in the neural model rather than an external knowledge source. In Gulcehre et al. (2016) a pointer model was used that can perform both copying and dictionary lookup. None of these studies address the problem of translation ambiguities resulting from added external knowledge sources. In truly low-resource languages, a dictionary obtained from the parallel training data will not have sufficient information to translate OOVs in the test data, as most of these will never have occurred in the training data. External dictionaries could be used in this case, which however requires a principled method of choosing between different translations.

An alternative strategy to address the rare and unknown word problem is to use subword units, i.e., the original text is segmented into chunks of characters, individual characters, or byte sequences. In Chung et al. (2016), a pure character-level decoder is used while Luong and Manning (2016) use a mixed model where the word-level decoder can fall back on the character-level decoder. The byte-pair encoding (BPE) approach of Sennrich et al. (2015) segments the input text into subword units based on an iterative merging of frequent character n-grams and a fixed upper size of the subword inventory. The main motivation given for the subword unit approach is that often a transparent relationship exists between OOVs and other known words: compound words and morphological variants can benefit from substantial overlap with other words in the same language, and cognates and named entities benefit from cross-lingual overlap. However, in resource-poor settings a substantial percentage of OOVs has no overt relationship with other words; instead, genuinely novel translations must be produced for words that were never seen and that are unrelated to other words.

A third class of approaches tries to leverage cognates and lexical borrowing. Tsvetkov and Dyer (2015) show that OOV words in low-resource languages that are loan words from a high-resource language can be translated via the high-resource language. However, the translation of OOV words in that work uses a fixed lexicon, not taking possibly multiple candidates into consideration. Finally, other studies have tried to exploit additional monolingual data in the source and/or target language. In Irvine and Callison-Burch (2013) new translation pairs were induced from monolingual corpora based on predictive features such as document timestamps, topic features, word frequency, and orthographic features. Saluja et al. (2014) and Zhao et al. (2015) explored the possibility of extracting features from extra monolingual corpora to help cover untranslated phrases. Specifically, Saluja et al. (2014) induced new translation rules from monolingual data with a semi-supervised algorithm. Zhao et al. (2015) obtained translation rules for OOV words based on phrases with similar continuous representations for which a translation is known.

Most of the studies described above have focused on neural MT for language pairs with sufficient training data. Recent work on OOV translation for low-resource languages includes Gujral et al. (2016), where a combination of approaches (surface and word-embedding based word similarity, transliteration) is used to generate multiple translation candidates for each OOV to improve phrase-based MT. The choice of a particular translation is then made either by a target-side language model or by the translation model itself through a secondary phrase table enriched with OOV-specific features.

3 OOV Disambiguation With Context Models

Our goal is to facilitate the integration of externally generated translation candidates, such as translation dictionaries, by utilizing a larger amount of target-side context information. We adopt a second-pass lattice rescoring approach that is compatible with both phrase-based and neural MT systems (or their combination) and that can accommodate extra monolingual information without increasing the number of parameters of the MT system itself. OOVs in the MT system's output are expanded to all translation options of that word found in our external knowledge sources. Target-language context models, possibly including context from beyond the current sentence boundaries, are then used to assign a score to each possible path in the extended lattice representing a particular combination of OOV translation hypotheses (see Figure 1).

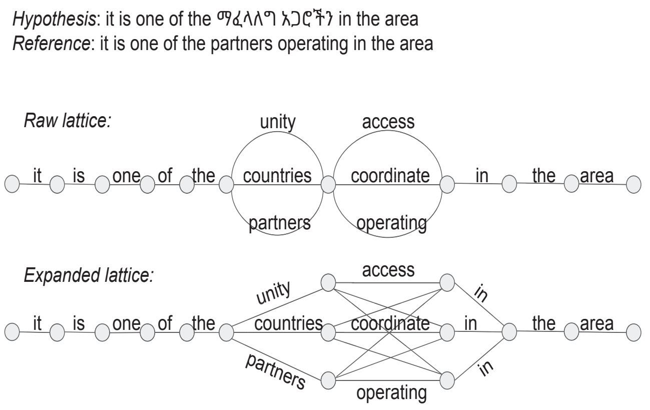

Figure 1: Compressed and expanded lattice representations of an MT hypothesis enriched with candidate translations of OOV words.

3.1 Count-Based Models

We compare several different models for rescoring, the simplest of which is a *sentence completion* model. OOV word translation can be formulated as sentence completion problem, where contextual information informs the filling of blanks in a sentence. Gubbins and Vlachos (2013) proposed to use a language model over a dependency tree for this task, whereas Woods (2016) and Röder et al. (2015) measure the degree of association between candidate words and other parts of the sentence using mutual information. In the same spirit our model chooses one out of several possible translation options for each OOV slot in the lattice based its average point-wise mutual information (PMI) with surrounding content words in the sentence (stopwords are ignored). PMI between words x and y is computed as:

$$PMI(x,y) = log\frac{P(x,y)}{P(x)P(y)} \tag{1}$$

The algorithm chooses one word at a time, proceeding from left to right. The chosen translation becomes part of the context used for computing PMI for the next set of OOV words. Therefore, the entire space of possible combinations of OOV translations is never fully explored but only searched greedily from left to right. Moreover, this method only focuses on content words and ignores word order in the PMI computation.

The second model is a graph-based reranking model (Mihalcea (2005); Yang and Kirchhoff (2012)), where an undirected graph is built over all OOV translation options and content words in a sentence. Graph edges are weighted with the same PMI values as in the sentence completion approach. PageRank (Page et al. (1999)) is then used to score each option based on 'votes' from connected words, and for each OOV slot, the options with the highest score is chosen. The PageRank score is computed as

$$R(v_i) = (1-d) + d \cdot \sum_{j \in IN(v_i)} \frac{R(v_j)}{|OUT(v_j)|} \tag{2}$$

where v is a vertex in the graph, d is a damping factor, and $IN()$ and $OUT()$ are in-degree and out-degree of the vertex, respectively. The difference to the sentence completion approach is that the entire space of translation combinations is explored globally rather than greedily. The reason we use the two above methods as baselines is that they can be trained cheaply and readily allow the integration of a larger document context by simply extending the list of content words with words from the previous or following sentences.

3.2 Neural Models

We next turn to neural models. At the sentence-level we utilize a recurrent neural language model, specifically a two-layer long-short term memory (LSTM) model. In order to extend the sentence-level LSTM to include information from previous sentences we follow the approach in Ji et al. (2015), which proposed several variants of document-context language models (DCLMs). Here, we use an attentional DCLM, which enriches a standard recurrent neural network with a context vector aggregating the hidden vectors in the previous sentence. A standard RNN computes the probability over output classes as

$$\mathbf{y}_{s,n} = softmax(\mathbf{W}_o \mathbf{h}_{s,n} + \mathbf{b}) \tag{3}$$

where s is the current sentence, n is the current time step in the sentence, $\mathbf{h}$ is a hidden vector, $\mathbf{b}$ is a bias vector, and $\mathbf{W}$ is a weight matrix. The hidden vector $\mathbf{h}_{s,n}$ is computed as

$$\mathbf{h}_{s,n} = g(\mathbf{h}_{s,n-1}, \mathbf{x}_{s,n}) \tag{4}$$

where g is a non-linear function (in this case, representing a two-layer LSTM) and x is an input vector (current word embedding). The attentional DCLM adds a context vector $c_{s,n}$ to both the hidden and the output layer as follows:

$$\mathbf{h}_{s,n} = g(\mathbf{h}_{s,n-1}, [\mathbf{x}_{s,n}^T, \mathbf{c}_{s-1,n}^T]^T) \tag{5}$$

$$\mathbf{y}_{s,n} = softmax(\mathbf{W}_o tanh(\mathbf{W}_h \mathbf{h}_{s,n} + \mathbf{W}_c \mathbf{c}_{s-1,n} + \boldsymbol{b})) \tag{6}$$

The context vector c is a position-dependent weighted linear combination of all hidden states $1, ..., M$ in the previous sentence.

$$\mathbf{c}_{s-1,n} = \sum_{m=1}^{M} \alpha_{m,n} \mathbf{h}_{s-1,m} \tag{7}$$

The attention weights are computed as

$$a_{n,m} = w_a^T tanh(\mathbf{W}_{a1} \mathbf{h}_{s,n} + \mathbf{W}_{a2} \mathbf{h}_{s-1,m}) \tag{8}$$

$$\alpha_n = softmax(a_n) \tag{9}$$

The attention weights $a_{n,m}$ encode the importance of each word in the previous sentence for the current word. DCLMs were shown to obtain reductions in perplexity compared to standard and hierarchical recurrent language models (Ji et al. (2015)); however, they were also observed to have a tendency towards overfitting when training data is sparse (Kirchhoff and Turner (2016)).

A different issue in applying neural language models to lattice rescoring is that each path in the lattice defines a different context; however, it is computationally infeasible to exhaustively rescore all paths. The number of OOV words per sentence is typically 3-5 in our tasks, and the number of translation candidates per word may go up to 30. In standard back-off n-gram models, lattice paths are merged based on identical truncated word histories, but this options is not available to us in neural language models where each hidden state encodes the cumulative untruncated history. Inspired by sentence-level lattice rescoring techniques explored in speech recognition (Liu et al. (2016)), we utilize a document-level lattice rescoring procedure that merges lattice paths based on the similarity of hidden state vectors in the model. The main steps are:

1. Start depth-first lattice traversal from the initial node $< s >$.
2. Use the context matrix $\boldsymbol{c}_{s-1}$ from the previous sentence initialize the hidden representation of the first word $< s >$.
3. At each lattice node, compute the hidden representation and the posterior probability of the word on the incoming arc according to a DCLM.
4. If there is another lattice path that shares the last word with the current lattice path, and in addition, if the hidden representations of these words fall below some distance threshold γ, then merge the two paths and update the probability and the hidden representation of the frontier word in the merged lattice path.

A detailed description of the algorithm is provided below in Algorithm 1. As a distance measure

Algorithm 1 Document-level lattice rescoring

```
1:  for each sentence $S$ in document $D$ do
2:      $L \leftarrow len(S)$
3:      for each node $n_i$ in the lattice do
4:          initialize its expanded node list $N_i = []$
5:          initialize its expanded outgoing arc list $A_i = []$
6:      $N_0 \leftarrow [n_0'^0]$
7:      $A_0 \leftarrow [a_{01}'^0, a_{01}'^1, ...]$
8:      for each expanded node $n_i'^j \in N_i$ do
9:          create outgoing arcs $a_{i,i+1}'^0, a_{i,i+1}'^1, ...$ according to translation candidates at node $n_{i+1}$
10:         for each outgoing arc $a_{i,i+1}'^k \in A_i$ do
11:             create expanded node $n_{i+1}'^k$
12:             $h_{i,i+1}^k \leftarrow$ hidden representation of the DCLM at $a_{i,i+1}'^k$
13:             $Pr(a_{i,i+1}'^k | a_{i-1,i}'^k, ...) \leftarrow$ posterior probability of the lattice path at $a_{i,i+1}'^k$
14:             if $\exists a_{i-1,i}'^l \in A_{i-1}$ and $a_{i-1,i}'^k = a_{i-1,i}'^l$ and $d(h_{i,i+1}^k, h_{i,i+1}^l) < \gamma$ then
15:                 if $Pr(a_{i,i+1}'^k | a_{i-1,i}'^k, ...) > Pr(a_{i,i+1}'^l | a_{i-1,i}'^l, ...)$ then
16:                     delete $n_{i+1}'^l$
17:                     prune the lattice branch that leads to $n_{i+1}'^l$
18:                 else
19:                     delete $n_{i+1}'^k$
20:                     prune the lattice branch that leads to $n_{i+1}'^k$
21:     Backtrack from the expanded node $n_L'^j \in N_L$ lattice path that has the highest probability to obtain the decoded sentence.
```

we use Euclidean distance between the hidden state vectors. The merging step is illustrated in

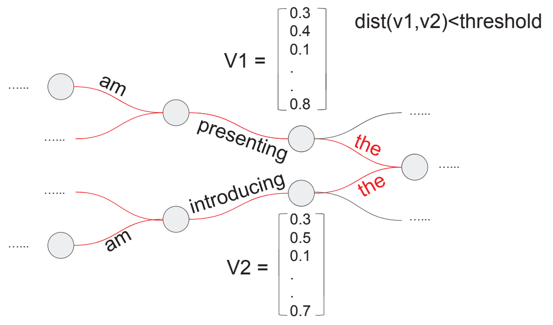

Figure 2: **Lattice path merging.** Paths ending in states whose associated hidden vectors are within a threshold distance of each other are merged.

Figure. 2. In practice, the procedure can be made computationally efficient by using a cache that maps each explored word to its (possibly multiple) hidden representations and posterior probabilities. In order to find the best path at the end of the traversal, the algorithm looks at all the remaining paths in the cache, finds the one that has the highest log-probability according to the context model, and traces back to the beginning of the path for the entire translation of this sentence. Also, the context matrix for this best path is propagated from the cache to the next sentence in the document for rescoring.

4 Data and Systems

4.1 MT Training Data

Our experiments are conducted on corpora for six different source languages (Amharic (amh), Uighur (uig), Somali (som), Yoruba (yor), Vietnamese (vie) and Hausa (hau)), with English as the target language. The corpora were provided as part of the DARPA LORELEI project on low-resource human language technology. Details of the training, development and test set sizes are provided in Table 1. Vocabulary sizes and OOV rates are shown in Table 3.

4.2 Context Model Training Data

The training data for the PMI-based context models consists of 4,264,684 Wikipedia articles[1]. PMI was computed based on the method and implementation described in Röder et al. (2015)[2]. The training data for each DCLM was selected from the Wiki-103 data set (Merity et al. (2016)), a collection of 28,475 Wikipedia articles (103M tokens) specifically curated for document-level language modeling. Data was selected separately for each language pair. For each article, a measure of overlap (Jaccard index) was computed between the article's vocabulary and the combined vocabulary extracted from the dev set target references and the one-best MT hypotheses on the test set. All articles with an index higher than 0.1 were included in the language model

[1] Wikipedia dump of 07/30/2014
[2] https://github.com/dice-group/Palmetto

	amh	uig	som	yor	hau	vie
train	63,181 / 5,941 / 1,237,172	99,005 / 18 / 2,587,335	52,288 / 8,872 / 1,097,616	41,525 / 7,538 / 933,932	43,370 / 3,554 / 423,935	28,686 / 439 / 423,069
dev	992 / 158 / 23,085	686 / 3 / 5,156	1,054 / 252 / 23,113	1,060 / 228 / 23,080	957 / 135 / 25,782	1,802 / 24 / 25,730
test	511 / 90 / 11,484	347 / 3 / 2,570	552 / 135 / 11,504	594 / 139 / 11,560	443 / 80 / 4,263	196 / 7 / 4,227

Table 1: Number of sentence pairs/documents/target language words in the training, development and test sets for each language.

training data. The same data was used for training the sentence-level LSTMs, to be able to directly compare the effect of document-level vs. sentence-level context on OOV disambiguation.

4.3 Baseline MT Systems

Baseline MT systems were developed for these tasks using phrase-based MT and attention-based neural MT (the Transformer model of Vaswani et al. (2017))[3]. The PBMT system was trained using Moses (Koehn et al. (2007)) and uses a flat phrase-based model with a maximum phrase length of 7, a backoff 4-gram language model trained on the target side of the parallel training data, and a bidirectional reordering model. The log-linear weights were trained using minimum error rate training on the dev set. The Transformer model was trained using a shared byte-pair encoding, resulting in a subword vocabulary of 8,000 word pieces. The hyperparameters of the Transformer models were tuned on the development sets with respect to the number of layers, layer dimensionality, learning rate, and regularization (dropout). The best parameters turned out to be: dropout rate of 0.1 at all layers, a learning rate of 0.2, 2 layers in the encoder and 2 layers in the decoder, and a hidden layer dimensionality of 512. The beam size during decoding is 4. Baseline results are shown in Table 2. Scoring was done in a case-insensitive fashion against a single reference translation.

Previous studies of neural sequence-to-sequence models for resource-poor scenarios (e.g., Koehn and Knowles (2017)) have found that PBMT models performed significantly better on low-resource languages unless the NMT models were enriched with additional components, such as a lexical memory (Nguyen and Chiang (2017)). By contrast, we find that self-attention based neural MT model performs comparably to PBMT, without any modifications to the basic model. A major contributor to the performance of the NMT models is the segmentation induced by byte-pair encoding, which results in system that outperform PBMT systems in 4 out of 6 language pairs. With a word-based vocabulary, NMT underperforms PBMT in most cases. Not surprisingly, languages with rich concatenative morphology (Amharic, Uighur) seem to benefit most from the subword approach.[4]

[3]We used the implementation provided at `https://github.com/tensorflow/tensor2tensor`

[4]PBMT models trained on the segmented text yielded worse scores than either word-based PBMT or Transformer

Language	PBMT	Transformer	Transformer w/ BPE
amh	16.93/49.6	13.15/46.0	**17.41/51.3**
uig	7.27/37.8	11.33/41.9	**17.22/46.7**
som	23.22/57.9	20.56/54.9	**25.36/59.9**
yor	18.22/50.8	15.68/49.1	**19.22/51.4**
hau	**21.86/57.8**	18.61/54.7	21.06/56.4
vie	**25.17/55.6**	22.83/53.1	23.00/54.2

Table 2: BLEU/unigram precision on test sets for phrase-based MT (PBMT), Transformer model, and Transformer model with byte-pair encoding (BPE).

	amh	uig	som	yor	hau	vie
Vocab	149,797	25,875	102,539	54,072	44,834	17,267
OOV rate (type)/(token)	18.4%/ 8.8%	32.4%/ 17.2%	14.5%/ 4.2%	13.4%/ 3.1%	8.5%/ 1.7%	10.5%/ 6.9%
Coverage (type)/(token)	99.8% 91.2%	47.6% 82.8%	85.0% 95.8%	83.7% 96.9%	80.1% 98.3%	86.4% 90.3%
Accuracy	5.6%	10.5%	15.9%	8.3%	8.8%	22.7%
# Candidates	8.0	22.0	15.4	18.4	20.6	28.6

Table 3: Vocabulary sizes, OOV rates, coverage, accuracy of external translation sources, and average number of translation candidates per OOV word.

4.4 Translation Candidate Generation

We obtain translation candidates for OOV words from (a) a large collection of web-crawled bilingual lexicons (Rolston and Kirchhoff (2016)) and (b) translations projected from related languages through Levenshtein distance based retrieval of words similar in their orthographic form. While the former is a reliable source, the latter method may introduce noise. Table 3 shows the number of OOVs, the coverage obtained by our external sources, accuracy (i.e., percentage of OOVs that have a translation matching the reference translation), and the average number of translation candidates per OOV. For all language pairs except for Uighur (which is morphologically highly complex), at least 80% of all OOV words receive a translation; however, the accuracy is at most 26% (note, however, that only a single reference translation was available; thus, synonyms are not counted).

5 Experimental Results

As an additional baseline we integrate the externally derived translations by simply adding them to the parallel training data, i.e., each translation pair is treated as an additional 'sentence'. Results are shown in Table 4. In this scenario, each of the new translation pairs is seen only once and without context; the final translation choice is still made by the MT system that has

models, even when using a larger maximum phrase length.

	PBMT	Transformer
amh	17.01 / 50.9	17.32 / 50.5
uig	7.84 / 42.2	**20.66 / 51.4**
som	23.91 / 58.6	**25.45** / 59.6
yor	18.35/ 50.6	**19.87 / 51.9**
hau	**21.94** / 57.8	21.55 / 56.7
vie	25.15 / 55.4	22.54 / 52.4

Table 4: Results (BLEU/unigram precision) of adding external translations to parallel training data. Boldface numbers are improvements over the best baseline system from Table 2.

Model	amh	uig	som	yor	hau	vie
LSTM	115.0	111.7	110.4	122.5	116.4	103.4
DCLM	101.7	103.1	100.3	98.6	97.3	95.4

Table 5: Perplexities obtained by LSTM vs. DCLM on dev sets.

been trained on the parallel data only. Compared to the baseline results in Table 2, we observe only mild improvements, except for Uighur, where the improvement is more pronounced.

We next conduct a diagnostic experiment designed to evaluate the different context models. To this end we enrich the list of translation candidates for each OOV word with the reference translation, in order to determine to what extent the different models are able to recover the correct translation if it is present in the candidate list. For simplicity we run these experiments on the output of the PBMT system, which, unlike the NMT output, contains the location of OOV words. Translation lattices were constructed from the one-best MT hypothesis and OOV translation candidates. The PMI and Pagerank systems were trained as described in Section 3. For PageRank, both sentence-level and document-level versions were trained, where the document context was defined to include the previous 4 sentences. We compare these against a sentence-level LSTM and the attentional DCLM described in Section 3. The sentence-level LSTM is a unidirectional model with two hidden layers of dimensionality 48. The DCLMs have a hidden layer size of 48 and also utilize a context size of 4 sentences. Word embedding vectors in both types of language models are initialized randomly. Both LSTMs and DCLMs were trained with DyNet (Neubig et al. (2017)).[5] The vocabulary for the language models consists of the OOV translation candidates and the words from the one-best MT hypotheses. A comparison between sentence-level and document-level model perplexities on the dev sets for each language pair is shown in Table 5.

The lattice rescoring results from the diagnostic experiment (Table 6) show that the attentional DCLM generally works best. The remaining experiments therefore use this model only.

We next apply the DCLM based rescoring method to our best baseline system, i.e., the Transformer system with BPE. Since this system decomposes all words into word pieces, it is not obvious which part of the output corresponds to an original OOV. We therefore align the PBMT and Transformer outputs, retain only the Transformer output and the aligned OOV slots, and replace OOV slots with their external translation options. We used fastAlign (Dyer et al.

[5]https://github.com/clab/dynet/tree/v1.1

	amh	uig	som	yor	hau	vie
PMI sent	16.68 / 49.3	7.82 / 37.8	22.56 / 57.4	17.91 / 49.6	20.89 / 57.3	25.87 / 55.8
PageRank sent	17.00 / 50.7	7.81 / 38.2	23.11 / 58.5	18.14 / 50.2	21.55 / 57.9	25.93 / 55.8
PageRank doc	16.97 / 50.8	8.04 / 38.9	**23.13** / 58.6	18.17 / 50.2	21.55 / 57.9	25.95 55.9
LSTM	17.01 / 50.7	9.91 / 49.0	22.98 / 59.1	17.87 / 50.2	21.21 / 57.7	25.97 / 56.0
DCLM	**17.39 / 52.9**	**9.96** / **50.4**	23.03 / **59.4**	**18.60 / 50.8**	**22.21 57.9**	**26.19 / 56.9**

Table 6: BLEU/unigram precision for lattice rescoring of PBMT output with reference translation included (diagnostic experiment).

(2013)) for this procedure, treating the Transformer output as source and the PBMT output as target for amh, uig, som, and yor. For vie and hau, we use the PBMT system's output as source since it outperforms the Transformer model. The results are shown in Row 3 in Table 7.

With the exception of Uighur, we find that our method slightly but consistently outperforms systems that utilize the external translations as additional training data (Row 2), indicating that contextual information is useful. For further calibration of the results we also provide topline results from an oracle experiment where the correct reference translation was substituted for every OOV slot (Row 4) – these numbers indicate the maximum possible improvement in BLEU/unigram precision that can be obtained from OOV translation on these tasks. For completeness we also provide the original baseline system scores (Row 1) and results obtained from full system combination (e.g., both the PBMT and the NMT's outputs are represented in the rescoring lattice, in addition to OOV translation options). Not surprisingly, system combination adds further improvements (except for Uighur), in some cases bringing the overall performance close to the topline. An example of the different system outputs is shown below:

	Method	amh	uig	som	yor	hau	vie
1	No OOV rescoring	17.41/ 51.3	17.22/ 46.7	25.36/ 59.9	19.22/ 51.4	21.86/ 57.8	25.17/ 55.6
2	Add'l train data	17.32/ 50.5	20.66/ 51.4	25.45/ 59.6	19.87/ 51.9	21.94/ 57.8	25.15/ 55.4
3	OOV rescoring	17.76 / 53.8	17.33 / 47.1	25.50 / 60.2	19.97 / 52.8	22.42 / 59.9	27.25 / 57.7
4	OOV topline	18.62 / 58.4	21.48 / 60.5	27.57 / 64.4	21.40 / 56.8	22.77 / 61.5	28.61 / 59.4
5	sys. comb.	18.24 / **56.1**	18.10 / 50.9	**27.00 / 63.0**	**20.82 / 55.5**	**22.65 / 60.7**	**28.19 / 58.5**

Table 7: BLEU/unigram precision of (1) baseline system without OOV handling; (2) systems trained with external translations as additional training data; (3) lattice rescoring with context models; (4) oracle; (5) full system combination of PBMT and Transformer outputs plus OOV translation.

Source sentence (with OOVs in italics):
saraakiisha ayaa sheegaya in qaraxu ka dhacay meel u dhow *albaadka* aqalka baarlamaanka , kaddib markii ilaaladu ay rasaas ku furtay *baabuurkaasi*
No oov rescoring:
officials said that the explosion took place near the parliament albaadka, after they opened fire on baabuurkaasi
Transformer output:
officials say that the explosion occurred near the house of the parliament after the guards opened fire on that vehicle
After rescoring with context model (our method):
officials said that the explosion took place near the parliament entrance, after they opened fire on kondoo
System combination:
officials said that the explosion occurred near the entrance of the parliament after the guards opened fire on that vehicle
Reference:
officials said the explosion took place near the entrance of the parliament building when guards opened fire on the vehicle

The best baseline system (Transformer with BPE) was able to correctly handle *baabuurkaasi* (vehicle) but not *albaddka* (entrance), which the rescoring procedure corrected. While this procedure also introduces an incorrect word (*kondoo*), rescoring of the lattice representing both the PBMT and Transformer output (system combination) in addition to OOV translations results in the correct output.

6 Conclusion

We have presented an approach towards the resolution of ambiguous translations of OOV words that arise when adding word translation pairs from external knowledge sources to an MT system. Of the different context models proposed, document-context language models with a context including previous sentences were shown to be most effective at identifying the correct translations. Our method showed substantial gains over baseline systems without special OOV handling and small but consistent gains over adding external translations directly to the training data, in five out of six language pairs. Future work will be concerned with integrating external resources and contextual information directly into neural MT architectures.

Acknowledgements
This paper was funded by DARPA under cooperative agreement no. HR0011-15-2-0043. The content of the information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

References

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Chung, J., Cho, K., and Bengio, Y. (2016). A character-level decoder without explicit segmentation for neural machine translation. *arXiv preprint arXiv:1603.06147*.

Dyer, C., Chahuneau, V., and Smith, N. A. (2013). A simple, fast, and effective reparameterization of ibm model 2. Association for Computational Linguistics.

Gubbins, J. and Vlachos, A. (2013). Dependency language models for sentence completion. In *EMNLP*, volume 13, pages 1405–1410. Citeseer.

Gujral, B., Khayralla, H., and Koehn, P. (2016). Translation of unknown words in low-resource languages. In *Proceedings of AMTA*.

Gulcehre, C., Ahn, S., Nallapati, R., Zhou, B., and Bengio, Y. (2016). Pointing the unknown words. *arXiv preprint arXiv:1603.08148*.

Irvine, A. and Callison-Burch, C. (2013). Supervised bilingual lexicon induction with multiple monolingual signals. In *Proceedings of HLT-NAACL*, page 518–523.

Jean, S., Cho, K., Memisevic, R., and Bengio, Y. (2014). On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*.

Ji, Y., Cohn, T., Kong, L., Dyer, C., and Eisenstein, J. (2015). Document context language models. *arXiv preprint arXiv:1511.03962*.

Kirchhoff, K. and Turner, A. M. (2016). Unsupervised resolution of acronyms and abbreviations in nursing notes using document-level context models. *EMNLP 2016*, page 52.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., 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.

Koehn, P. and Knowles, R. (2017). Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver. Association for Computational Linguistics.

Liu, X., Chen, X., Wang, Y., Gales, M., and Woodland, P. (2016). Two efficient lattice rescoring methods using recurrent neural network language models. volume 24(8).

Luong, M.-T. and Manning, C. D. (2016). Achieving open vocabulary neural machine translation with hybrid word-character models. *arXiv preprint arXiv:1604.00788*.

Luong, M.-T., Sutskever, I., Le, Q. V., Vinyals, O., and Zaremba, W. (2014). Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206*.

Merity, S., Xiong, C., Bradbury, J., and Socher, R. (2016). Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Mihalcea, R. (2005). Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling. In *Proceedings of EMNLP*.

Neubig, G., Dyer, C., Goldberg, Y., Matthews, A., Ammar, W., Anastasopoulos, A., Ballesteros, M., Chiang, D., Clothiaux, D., Cohn, T., Duh, K., Faruqui, M., Gan, C., Garrette, D., Ji, Y., Kong, L., Kuncoro, A., Kumar, G., Malaviya, C., Michel, P., Oda, Y., Richardson, M., Saphra, N., Swayamdipta, S., and Yin, P. (2017). Dynet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980*.

Nguyen, T. Q. and Chiang, D. (2017). Improving lexical choice in neural machine translation. *arXiv preprint arXiv:1710.01329*.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab.

Röder, M., Both, A., and Hinneburg, A. (2015). Exploring the space of topic coherence measures. In *Proceedings of the eighth ACM international conference on Web search and data mining*, pages 399–408. ACM.

Rolston, L. and Kirchhoff, K. (2016). Collection of bilingual data for lexicon transfer learning. Technical Report UW-EE-2016-0001.

Saluja, A., Hassan, H., Toutanova, K., and Quirk, C. (2014). Graph-based semi-supervised learning of translation models from monolingual data. In *ACL (1)*, pages 676–686.

Sennrich, R., Haddow, B., and Birch, A. (2015). Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Tsvetkov, Y. and Dyer, C. (2015). Lexicon stratification for translating out-of-vocabulary words. In *ACL (2)*, pages 125–131.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. arXiv:1706.03762.

Woods, A. M. (2016). Exploiting linguistic features for sentence completion. In *The 54th Annual Meeting of the Association for Computational Linguistics*, page 438.

Yang, M. and Kirchhoff, K. (2012). Unsupervised translation disambiguation for cross-domain statistical machine translation. In *Proceedings of association for machine translation in the Americas*.

Zhao, K., Hassan, H., and Auli, M. (2015). Learning translation models from monolingual continuous representations. In *HLT-NAACL*, pages 1527–1536.