Learning Translation Models from Monolingual Continuous Representations

Kai Zhao*
Graduate Center, CUNY
New York, NY 10016, USA
kzhao.hf@gmail.com

Hany Hassan
Microsoft Research
Redmond, WA 98502, USA
hanyh@microsoft.com

Michael Auli*
Facebook AI Research
Menlo Park, CA 94025, USA
michaelauli@fb.com

Abstract

Translation models often fail to generate good translations for infrequent words or phrases. Previous work attacked this problem by inducing new translation rules from monolingual data with a semi-supervised algorithm. However, this approach does not scale very well since it is very computationally expensive to generate new translation rules for only a few thousand sentences. We propose a much faster and simpler method that directly hallucinates translation rules for infrequent phrases based on phrases with similar continuous representations for which a translation is known. To speed up the retrieval of similar phrases, we investigate approximated nearest neighbor search with redundant bit vectors which we find to be three times faster and significantly more accurate than locality sensitive hashing. Our approach of learning new translation rules improves a phrase-based baseline by up to 1.6 BLEU on Arabic-English translation, it is three-orders of magnitudes faster than existing semi-supervised methods and 0.5 BLEU more accurate.

1 Introduction

Statistical translation models (Koehn et al. 2003, Chiang et al. 2005) are trained with bilingual data and a simple solution to improve accuracy is to train on more data. However, for many language pairs we only have a very limited amount of bilingual data and even when dealing with resource-rich languages, we still often perform poorly when dealing with rare words or phrases.

On the other hand, there is plenty of monolingual data and previous work has investigated its use in learning translation models (Rapp, 1995; Callison-Burch et al., 2006; Haghighi et al., 2008; Saluja et al., 2014). However, most methods rely on statistics that are computationally expensive. As a concrete example, the graph propagation algorithm of Saluja et al. (2014) relies on pair-wise mutual information statistics between any pair of phrases in the monolingual corpus that is very expensive to compute, even for moderately sized corpora.

In this paper, we study the use of standard continuous representations for words to generate translation rules for infrequent phrases (§2). We explore linear projections that map continuous representations of rare foreign phrases to English phrases. In particular, we propose to learn many local projections that are specific to a given foreign phrase. We find this to be much more accurate than a single globally learned mapping such as proposed by (Mikolov et al. 2013; §3).

Our method relies on the fast retrieval of similar phrases in continuous space. We explore both Locality Sensitive Hashing (LSH; Indyk and Motwani, 2008) as well as the lesser known Redundant Bit Vector method (RBV; Goldstein et al. 2005) for fast k-nearest neighbor (k-NN) search. RBV outperforms the popular LSH algorithm by a large margin, both in speed as well as accuracy (§4).

Our results show that the local linear projection method is not only three orders of magnitudes faster than the algorithm of Saluja et al. (2014) but also by 0.5 BLEU more accurate. We achieve a 1.6 BLEU improvement in Arabic-English translation compared to a standard phrase-based baseline (§5).

2 Continuous Phrase Representations

Continuous representations of words have been found to capture syntactic and semantic regularities in language (Turian et al., 2014; Collobert et al., 2011; Mikolov et al., 2013c). The induced representations often tend to cluster similar words together as illustrated in Figure 1.

*The entirety of this work was conducted while at Microsoft Research.

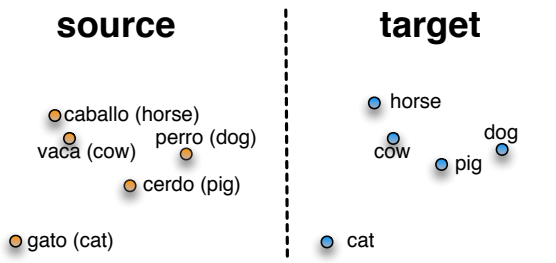


Figure 1: Illustration of word representations in Spanish and English (Figure from Mikolov et al. (2013a)). The plots are based on a two-dimensional projection of the original vectors with principal component analysis.

A logical next step is to learn representations for larger linguistic units, a topic which has received a lot of interest (Mitchell and Lapata, 2010; Socher et al., 2011; Le and Mikolov, 2014). For machine translation there have been efforts to learn representations for entire bilingual phrases (Zou et al., 2013; Zhang et al., 2014; Gao et al., 2014).

In this work, we only require representations for monolingual phrases that are relatively short.[1] We therefore decided to use off-the-shelf word representations to build phrase vectors. In particular, we chose the continuous bag-of-words model (Mikolov et al., 2013b) which is very fast to train and scales very well to large monolingual corpora.

The resulting word vectors are then used to build phrase vectors via simple element-wise addition which has been found to perform very competitively in comparison to alternative approaches (Mitchell and Lapata, 2010). Note that all the algorithms described in this paper are agnostic to the choice of phrase-representation and other schemes may perform better.

We use these monolingual phrase representations to generate translation rules for infrequent, or *unlabeled*, phrases. Unlabeled phrases do not appear in the bilingual data and thus do not have translation rules. The general idea behind the following algorithms is to identify *labeled* phrases for which we know translation rules that are similar to an unlabeled phrase, and to use them to induce translation rules for the unlabeled phrase.

[1]For simplicity, we only consider unigrams and bigrams on the source side, see §5

3 Translation Rule Generation

We first describe how we can learn a single mapping between the foreign and English continuous spaces to find translations for an infrequent foreign phrase (§3.1). Next, we make this approach more robust by learning many mappings that are specific to a given foreign phrase (§3.2). Finally, we review the semi-supervised label propagation algorithm of Saluja et al. (2014) which we make much faster using continuous word representations and k-NN algorithms (§3.3).

3.1 Global Linear Projection

Mikolov et al. (2013a) find that the relative positions between words are preserved between languages (Figure 1), and, thus, it is possible to learn a *linear* projection that maps the continuous representation of source phrases to points on the target side. The hope is to learn a mapping that captures the relationship between the source and target spaces. We call this linear transform *global linear projection*, since we use a single mapping that we apply to every source phrase.

More formally, we denote f and e as source side and target side phrases respectively, and $\mathbf{f} \in \mathbb{R}^{1\times d}$ and $\mathbf{e} \in \mathbb{R}^{1\times d}$ as the corresponding phrasal vectors with dimension d. Following Mikolov et al. (2013a), we learn a global linear projection matrix $W \in \mathbb{R}^{d\times d}$ based on the translations of the n most frequent labeled source side phrases: $(\mathbf{f}_1, \mathbf{e}_1), (\mathbf{f}_2, \mathbf{e}_2), \ldots, (\mathbf{f}_n, \mathbf{e}_n), n \geq d$.[2] Let $F = [\mathbf{f}_1^T, \mathbf{f}_2^T, \ldots, \mathbf{f}_n^T]^T$, and $E = [\mathbf{e}_1^T, \mathbf{e}_2^T, \ldots, \mathbf{e}_n^T]^T$. We calculate W by solving the following linear system:

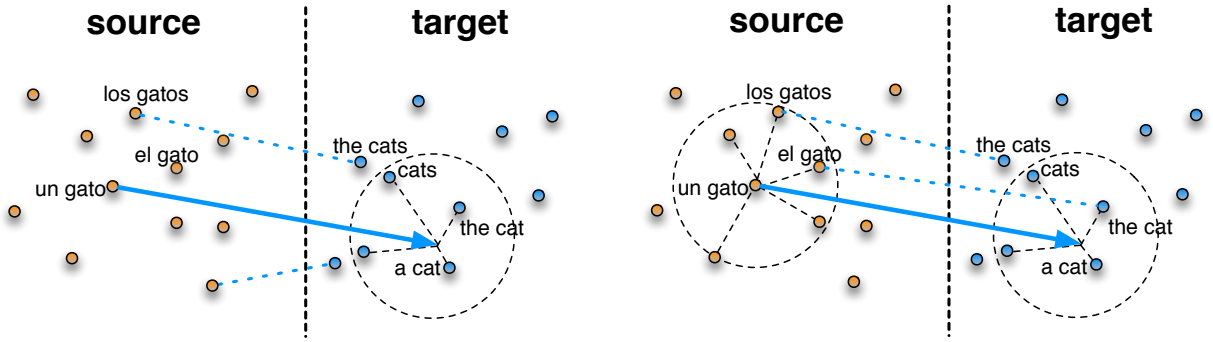
$$FW = E$$

whose solution is:

$$W \approx (F^T F)^{-1} F^T E$$

Using the linear transform W, we can compute $\bar{\mathbf{e}} = \mathbf{f}W$ for each unlabeled source phrase $\mathbf{f}$, where $\bar{\mathbf{e}}$ will be close to target phrases that are potential translation candidates for $\mathbf{f}$. We denote the set of all nearby English phrase vectors as $N(\bar{\mathbf{e}})$ and use

[2]We need more than d phrases to be fetched to make the linear system solvable. Similar is for the local linear projection in §3.2.



(a) Global Linear Projection

(b) Local Linear Projection

Figure 2: (a) Illustration of the global linear projection mapping the unlabeled Spanish phrase "un gato" to the target space. The neighbors of the projected point serve as translation candidates and are fetched via a k-NN query. (b) A local linear projection is learned individually for "un gato" based on the translations "the cats", "the cat" of the labeled neighbors "los gatos", "el gato".

fast k-NN query algorithms to retrieve this set (§4). Figure 2 (a) illustrates the method.

The translation probability for each translation candidate $\mathbf{e} \in N(\bar{\mathbf{e}})$ is based on the similarity to the projected point $\bar{\mathbf{e}}$:

$$P(e|f) = \frac{\exp\{\text{sim}(\mathbf{e}, \bar{\mathbf{e}})\}}{\sum_{\mathbf{e}' \in N(\bar{\mathbf{e}})} \exp\{\text{sim}(\mathbf{e}', \bar{\mathbf{e}})\}}. \quad (1)$$

Note that we normalize over the neighbor set $N(\bar{\mathbf{e}})$ of the projected point $\bar{\mathbf{e}}$ of foreign phrase f. This uses the similarity $\text{sim}(\bar{\mathbf{e}}, \mathbf{e})$ between $\bar{\mathbf{e}}$ and $\mathbf{e}$ which is defined symmetrically as

$$\text{sim}(\bar{\mathbf{e}}, \mathbf{e}) = \frac{1}{1 + \|\bar{\mathbf{e}} - \mathbf{e}\|}, \quad (2)$$

where $\|\bar{\mathbf{e}} - \mathbf{e}\|$ is the Euclidean distance between vectors $\bar{\mathbf{e}}$ and $\mathbf{e}$.

Before adding the generated candidate translations to the MT system, we also calculate the backward maximum likelihood translation probability using Bayes' Theorem:

$$\text{P}(f|e) = \frac{\text{P}(e|f)\text{P}(f)}{\text{P}(e)},$$

where the marginal probabilities are based on the counts of phrases seen in the monolingual corpora.

Similar to Saluja et al. (2014), we use word-based translation probabilities from the baseline system to obtain forward and backward lexicalized translation probabilities.

3.2 Local Linear Projection

The global linear projection uses a single projection matrix for all unlabeled source phrases. This is simple and fast but assumes that we can capture all relations between the source and target representation space with a single $\mathbb{R}^{d \times d}$ mapping. We show later that this is clearly not the case (§5.4) and that a single projection struggles particularly with infrequent phrases - the precise situation in which we would like our projection to be robust.

We therefore propose to learn many *local linear projections* which are individually trained for each unlabeled source phrase. Specifically, for each unlabeled source phrase $\mathbf{f}$, we learn a mapping $W_{\mathbf{f}} \in \mathbb{R}^{d \times d}$ based on the translations of m of $\mathbf{f}$'s labeled neighbors: $(\mathbf{f}_1, \mathbf{e}_1), (\mathbf{f}_2, \mathbf{e}_2), \ldots, (\mathbf{f}_m, \mathbf{e}_m)$, $\mathbf{f}_i \in N(\mathbf{f})$, $1 \leq i \leq m$, $m \geq d$ (see Figure 2 (b)).

Compared to the global projection, we require an additional k-NN query to find the labeled neighbors for each unlabeled source phrase. However, this extra computation takes only a negligible amount of time, since the number of labeled phrases on the source side is significantly smaller than the number of phrases on the target side.

Our approach of learning many different mappings is similar to the *locally linear embedding* method of Roweis and Saul (2000) for nonlinear dimensionality reduction, which also uses neighboring points to construct a locally precise linear projection in order to map to another space.

3.3 Structured Label Propagation with Continuous Representation

Saluja et al. (2014) use Structured Label Propagation (SLP; Liu et al. 2012) to propagate candidate translations from frequent source phrases that are labeled to unlabeled neighbors that are infrequent.

The algorithm works as follows: for a known translation rule (f', e'), SLP propagates the target side phrases $e \in N(e')$, that are similar to e', to the unlabeled source phrases $f \in N(f')$, that are similar to f', as new translation rules. This propagation runs for several iterations. At each iteration, the translation probability between known translations is fixed. More formally, for iteration $t+1$ we have

$$\mathrm{P}^{t+1}(e|f) = \sum_{f' \in N(f)} T(f'|f) \sum_{e' \in H(f')} T(e|e') \mathrm{P}^{t}(e'|f'),$$

where $T(f'|f)$ is the probability that phrase f is propagated through phrase f', similarly for $T(e|e')$; $H(f')$ is the set of translation candidates for source phrase f', which is learned from the bilingual corpus.

In Saluja et al. (2014), both $T(f'|f)$ and $T(e|e')$ are based on the pairwise mutual information (PMI) between two phrases. Computing PMI statistics between any two phrases over a large corpus is infeasible and therefore the authors resort to a simple approximation that only considers co-occurrences with other phrases within a fixed-sized context window. Even after this simplification the running time of the SLP is vastly dominated by gathering similarity statistics and by constructing the resulting graph.

However, once the PMI statistics are collected and the graph is constructed, actual label propagation is very fast. To speed up the algorithm, we replace the costly PMI statistics by continuous phrase representations and adopt the same similarity measure that we used for the global and local projections (see Equation 1). Moreover, we replace the static graph construction with on-demand graph expansion using the fast phrase query mechanisms described in the next section. These modifications allow us to dramatically speed up the original SLP algorithm as demonstrated in our experiments (§5).

4 Fast Phrase Query with Continuous Representation

The algorithms presented in the previous section require rapid retrieval of neighboring phrases in continuous space. Linear search over all n candidate phrases is impractical, particularly for the SLP algorithm (§3.3). SLP requires the construction of a graph encoding the nearest neighbors for each target phrase, be it online or offline. To construct this graph naïvely requires $\mathcal{O}(n^2)$ comparisons which is clearly impractical for our setup where we have over one million target phrases (§5). For the linear projections, we still need to run at least one k-NN query in the target space for each infrequent foreign phrase.

Various methods, e.g., k-d trees, were proposed for fast k-NN queries but most of them are not efficient enough in high dimensional space, such as our setting. We therefore investigate *approximated* k-NN query methods which sacrifice some accuracy for a large gain in speed. Specifically, we look into locality sensitive hashing (LSH; §4.1), a popular method, as well as redundant bit vectors (RBV; §4.2), which to our knowledge has not been previously used for natural language processing tasks.

4.1 Locality Sensitive Hashing

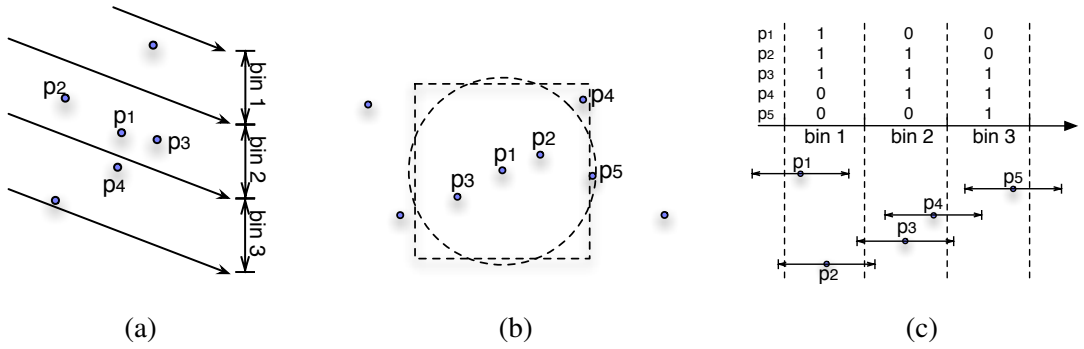
One popular approximated method is Locality Sensitive Hashing (LSH; Indyk and Motwani, 1998), which has been used in many NLP tasks such as noun clustering (Ravichandran et al., 2005), topic detection (Petrović et al., 2010), and fast k-NN query for similar words (Goyal et al., 2012).

For our particular task, assume each phrase is represented by a d-dimensional vector $\mathbf{p}$ of real values. The core of LSH is a set of hash functions. We choose p-stable distribution based functions (Datar et al., 2004) of the following form:

$$h_i(\mathbf{p}) = \lfloor \frac{\mathbf{x}_i \cdot \mathbf{p} + b_i}{w} \rfloor, 1 \leq i \leq s.$$

This function can be viewed as a quantized random projection, where each element in $\mathbf{x}_i$ is selected randomly from a Gaussian distribution $\mathcal{N}(0,1)$, w is the width of the bin, b_i is a linear bias selected from a uniform distribution $\mathcal{U}(0,w)$ (see Figure 3 (a)).

By concatenating the results from h_i, $1 \leq i \leq s$, phrase $\mathbf{p}$ is projected from d-dimensional space to



(a) (b) (c)

Figure 3: (a) A quantized random projection in LSH. The arrows show the direction of the projection. Points p_1, p_2, p_3 are correctly projected to the same bin, while p_4 falls into another bin, despite being very close to p_1. (b) A simplified example illustrating RBV in two dimensions. The circle with radius r is centered at $\mathbf{p}_1$ and contains all neighbors of $\mathbf{p}_1$. RBV approximates the circle by a square of width $d = 2 \times 0.95r$, which contains most of the neighbors of $\mathbf{p}_1$ but also $\mathbf{p}_4$, a false positive, while missing $\mathbf{p}_5$, a closer point. (c) On each dimension, RBV uses bit vectors to maintain the set of points whose hypercubes (represented as the segments on the points in 1-dimensional view) intersect with a bin.

an s-dimensional space. Phrases whose projections collide in the s-dimensional space are considered candidates to be neighbors. A fast retrieval of those colliding phrases can be done via a hash table. However, since the projection is random, it is very likely that true neighbors in the d-dimensional space fall into different bins after projection (false negatives; e.g., p_1 and p_4 in Figure 3 (a)). To ease this problem, LSH employs a set of such projections and runs a linear search over the *union* of all possible neighbor candidates resulting from these projections to find the approximated k-nearest neighbors.

4.2 Redundant Bit Vectors

The performance of LSH decreases as the number of dimensions grows. Redundant bit vectors (RBV; Goldstein et al., 2005) address this problem and can quickly search in high dimensional space, which suits our task better.

RBV is a combination of: a) an approximated neighborhood test designed for high dimensional space, and b) an efficient data structure for fast neighborhood query.

First, for a given point $\mathbf{p}$ in high dimensional space, the volume of a hypersphere of radius r centered at $\mathbf{p}$ can be approximately covered by a hypercube of width $d = 2\epsilon r$, $\epsilon \ll 1$.[3] Figure 3 (b) shows an illustration in two dimensional space where a square of width $d = 2 \times 0.95r$ covers most of a circle with radius r. In higher dimensional space, e.g., $d = 256$ as in Goldstein et al. (2005), we can cover 99% of the volume of a hypersphere of $r = 1$ with a hypercube whose width is only $\sim 2 \times 0.2r$.[4]

This surprising result allows us to use a very small hypercube to approximate the hypersphere. Furthermore, if two points $\mathbf{q}$ and $\mathbf{p}$ are within a certain radius r, i.e., $\|\mathbf{q} - \mathbf{p}\| \leq r$, then frequently $|\mathbf{q}^{(i)} - \mathbf{p}^{(i)}| \leq \epsilon r$, where $\mathbf{x}^{(i)}$ denotes the i-th element of vector $\mathbf{x}$. Thus, the neighbor query can be approximated as a check whether the distance between $\mathbf{p}$ and $\mathbf{q}$ on each dimension is less than ϵr, $\epsilon \ll 1$.

Second, each dimension is quantized into bins. Each bin *redundantly* maintains a set of points whose hypercubes intersect with the bin on that dimension. This set is an approximation of the neighbors of a query point $\mathbf{p}$ that falls into the same bin on this dimension. RBV uses *bit vectors* to store this set of points for each bin. (See Figure 3 (c).)

For a given query vector $\mathbf{p}$, we fetch the bins where $\mathbf{p}$ falls into for each dimension. We then per-

[3] Here we use $\ll$ in an imprecise way. $\epsilon \ll 1$ does not mean ϵ is smaller than 1 by orders of magnitudes; usually $\epsilon > 0.1$. What we mean is that in high dimensional space, the volume of a hypercube of width $2\epsilon r$ is more than hundreds of magnitudes smaller than a hypercube of width $2r$.

[4] Note that this does not mean the volume of the hypercube is smaller than the hypersphere. It just means that most of the volume of the hypersphere is covered in the hypercube.

	English	Arabic	Urdu
Token count	5b	5b	75m
Word vector count	2.9m	2.7m	0.2m
Word vector train time	100hrs	100hrs	3hrs

Table 1: Monolingual corpora statistics, number of word vectors, and time to learn the word vectors (on single CPU core) for each source language.

form a bitwise *and* over the corresponding bit vectors to find the set of points that actually fall into $\mathbf{p}$'s hypercube, i.e., the approximated candidate neighbor set of $\mathbf{p}$. Finally, a linear search over this much smaller set finds the approximate k-nearest neighbors, similar to LSH.

5 Experiments

We first evaluate the speed and accuracy of the presented approximate k-NN query algorithms (§5.2). Next we experiment with the translation rule generation approaches (§5.3), and then we analyze the global and local projection methods (§5.4). Following Saluja et al. (2014), we present most results on Arabic-English translation and then validate our findings on Urdu-English (§5.5), a low-resource setting. Lastly, we discuss some qualitative results (§5.6).

5.1 Datasets & Preprocessing

We test our approach on both Arabic-English and Urdu-English translation. For Arabic-English our bilingual training data comprises of 685k sentence pairs. The NIST MT08 and MT09 data sets serve as tuning and testing sets, respectively. Both are combinations of newswire and weblog articles, and each Arabic sentence has four reference translations. For Urdu-English our bilingual training corpus contains 165k sentence pairs, and the tuning and testing sets are NIST MT08 and NIST MT09, respectively.

Table 1 shows some statistics for the monolingual data we use. The majority of the data for Arabic and English is drawn from the AFP Gigaword corpus. For Urdu most of the data is mined by a web crawler, mainly because there are not many official resources for this language.

We run standard tokenization and segmentation on the monolingual corpora. After that we use the Word2Vec tool (Mikolov et al., 2013b) to generate word embeddings for each language with the bag-of-words model, where the number of dimensions is set to $d = 300$. See Table 1 for the number of word vectors learned for each language.

	False Negative (%)	Time (s)
Linear Search	0	342
LSH	14.29	69
RBV	9.08	19

Table 2: Performance of linear search, locality sensitive hashing, and redundant bit vectors, for $k = 200$.

To obtain phrases in each language, we use a similar strategy as in Saluja et al. (2014). For Arabic and Urdu, we collect all unigrams and bigrams from the tuning and testing sets. This gives 0.66m phrases for Arabic and 0.2m phrases for Urdu. For English, we collect unigrams and bigrams from the monolingual data instead. However, the English monolingual corpus is much larger than the tuning and testing sets for Arabic and Urdu. We therefore train a language model over the monolingual data, and collect the unigrams and bigrams from the ARPA file, filtering out all candidates that have a probability smaller than 10^{-7}. Similar to Saluja et al. (2014), we use a baseline MT system to translate the Arabic or Urdu phrases and add their translations to the English phrase set. After this procedure we end up with 1.5m English phrases.

We use simple component-wise addition to generate phrase vectors from word vectors. Some rare words do not receive a vector representation after running Word2Vec, and we simply remove phrases containing those words, resulting in a total of 0.65m phrases for Arabic, 0.18m phrases for Urdu, and 1.2m phrases for English.

5.2 Evaluation of Approximated k-NN Query

We first evaluate the performances of different k-NN query approaches on English word vectors.

There are 2.9m word vectors in $d = 300$ dimensional space. We randomly select 1,000 words, and query for each word the 200 nearest neighbors, $k = 200$, with either linear search, LSH, and RBV. We measure the false negative ratio, i.e., the percentage of true neighbors missed by each query method, as well as time. For LSH and RBV, we tune the parameters for best performance (LSH: number of projected dimensions, number of layers, and width of

	Tune	Test	Time (hr)
Baseline	39.33	38.09	-
SLP w/ PMI	40.93	39.16	$\sim$10,000
SLP w/ Cont. Repr.	41.31	39.34	120+200
GLP	40.46	38.68	20+200
LLP	41.17	39.57	30+200
LLP w/ backoff	41.48	39.70	30+200

Table 3: Arabic-English translation accuracy of structured label propagation with PMI (SLP) and with continuous representations (SLP w/ PMI), the global linear projection (GLP), our local linear projection (LLP) and with an added backoff scheme (LLP w/ backoff). For applicable methods, we list the running time to compute distributional representations as a separate term in the time column. This is usually only required once per language which is why we report it separately.

the bin; RBV: hypercube width and number of bins for each dimension).

Table 2 shows that RBV gives significantly better performance than LSH, both in terms of accuracy and speed. RBV reduces the false negative ratio by $1/3$ compared to LSH and is 3.6 times faster. This is in line with Goldstein et al. (2005) who observed that the performance of LSH degrades in high dimensional space. We therefore use RBV in the following experiments.

5.3 Evaluation of Rule Generation

Next, we evaluate the quality of the generated translation rules for Arabic-English translation (Table 3) using either SLP, the global linear projection (GLP), or the local linear projection (LLP).

Our baseline system is an in-house phrase-based system similar to Moses with a 4-gram language model. The underlying log-linear model comprises of 13 features: two maximum likelihood translation probabilities, two lexicalized translation probabilities, five hierarchical reordering model features (Galley and Manning, 2008), one language model, word penalty, phrase length, and distortion penalty), and is tuned with minimum error rate training (MERT; Och 2003). Translation quality is measured with BLEU (Papineni et al., 2002).

For comparison, we reimplemented the graph-based method in Saluja et al. (2014). This method calculates the pairwise mutual information (PMI) between phrases, and employs all the techniques mentioned in Saluja et al. (2014) to speedup the computations. Our reimplementation achieves similar performance to Saluja et al. (2014) (with a negligible $\sim$ 0.06 drop in BLEU). We parallelized the algorithm on a cluster since a single core implementation would run for $\sim$10k hours.[5]

Our continuous phrase based version of SLP is orders of magnitudes faster than the SLP variant of Saluja et al. (2014) because it replaces the computationally expensive PMI calculation by an approximated k-NN query in distributional space. Moreover, our variant of SLP even improves translation quality by 0.2-0.3 BLEU. Overall, our version of SLP improves the baseline by 2.0 BLEU on the tuning set and by 1.3 BLEU on the test set.

The linear projection based methods, GLP and LLP, are in turn again several times faster than SLP with continuous representations. This is because they require significantly fewer k-NN queries. For both GLP and LLP, we retrieve the 200 nearest neighbors of the projected point. For LLP, the local projection is calculated based on the 500 nearest labeled neighbors of the infrequent source phrase. LLP achieves slightly better accuracy on the test set than PMI-based SLP but at four times the speed. GLP is the fastest method but also the least accurate, improving the baseline only by about 0.6 BLEU. We explore this result in more detail in the next section. Overall, our local projection outperforms the global projection by 0.9 BLEU on the test set.

For some infrequent source phrases, approximated k-NN query does not retrieve enough ($\geq d$) neighbors to learn a local linear projection. For these phrases, we employ a *backoff* strategy that uses the translations of their neighbors as additional translation candidates. This strategy provides helpful additional rules for LLP (Table 3).[6]

5.4 Evaluation of Global Linear Projection

To learn why GLP does not generate high quality translation rules, we run an extra experiment to measure the projection quality of GLP.

We train a global linear projection on an increas-

[5]Confirmed with the authors of Saluja et al. (2014) from personal communication.

[6]The backoff scheme in the Arabic-English setting generates around 15% of the translations rules, which adds 0.13 BLEU on the test set. This is not a big improvement and so we did not employ this scheme for our Urdu-English experiments.

Training Set	Hit Rate: Freq	Hit Rate: Infreq.
500	0.87	0
1,000	0.6	0.01
5,000	0.42	0.07
25,000	0.4	0.05

Table 4: Quality of global linear projection measured by the ratio that GLP can fetch the most possible translation in the 200-nearest neighbors.

	Tune	Test	Time (hr)
Baseline	26.32	27.41	-
SLP w/ PMI	27.26	27.89	∼7,000
SLP w/ Cont. Repr.	27.34	27.73	100+103
LLP	27.06	27.98	30+103

Table 5: Urdu-English translation accuracy (cf. Table 3).

ing amount of training data and measure its accuracy on two test sets (Table 4). The first test set contains the 100 most frequent source phrases and their translations. The second test set contains less frequent examples; we choose the 50,000 to 50,100 most frequent source phrases. The training data uses the l most frequent source phrases and their translations which are not already contained in the first test. The projection quality is measured by the ratio of how many times the correct translation is one of the 200-nearest neighbors of the projected point computed by GLP.

The results in Table 4 clearly show that GLP can find the best translation for very frequent source phrases which is in line with previous work Mikolov et al. (2013a). However, the accuracy for infrequent phrases is poor. This explains why GLP helps relatively little in our translation experiments because our setup requires a method that can find good translations for infrequent source phrases.

5.5 Evaluation on Urdu-English

Resources for Urdu are limited compared to Arabic (§5.1) which results in fewer word vectors and fewer source phrases. This will also affect the quality of the word vectors in Urdu, since more training data usually results in better representations.

Table 5 shows that the improvements of both SLP and LLP in Urdu-English are not as significant as for Arabic-English. Our reimplementation of SLP is ~ 1 BLEU better on the tuning set than the baseline, and ~ 0.5 BLEU better on the test set. As expected, the translation quality improvement on small corpora is not as significant as on large corpora like Arabic, since the monolingual data in Urdu is much smaller than for Arabic (75m tokens vs. 5b tokens) which makes it more difficult to learn good representations. In general, with continuous representations, SLP and LLP achieve similar performance to PMI-based SLP but the projection based methods are orders of magnitudes faster.

Source	Generated target
التزاماتها الانسانيه	the humanitarian obligations
التزاماتها الانسانيه	humanitarian commitments
هاتين المجموعتين	both these two groups
هاتين المجموعتين	these two communities
بناء مؤسساتهم	building their institutions
کوششیں ضرور	certainly efforts
کوششیں ضرور	efforts must
مند نوجوانوں	healthier youth
سپیشل سروسز	services special
کمیونٹی ڈیولپمنٹ	community development

Figure 4: Examples of the generated rules from LLP.

5.6 Analysis of Output

Figure 4 shows some examples of the translation rules produced by our system. The first five examples are for the Arabic-English system, while the last five are for the Urdu-English system. All source phrases are unknown to the baseline system which usually results in sub-optimal translations. Our system on the other hand, managed to generate translation rules for them. The Arabic-English examples show mostly morphological variants of phrases which did not appear in the parallel data; this can be helpful for highly inflected languages since most of the inflectional variations are underrepresented in the parallel data. The Urdu-English examples show mostly unknown phrases since there is much less parallel data than for Arabic.

6 Conclusion and Future Work

In this work, we showed how simple continuous representations of phrases can be successfully used to induce translation rules for infrequent phrases and demonstrated substantial gains in translation accuracy. Continuous representations not only increase the speed of the semi-supervised approach of Saluja et al. (2014) by two orders of magnitude but also improve its accuracy at the same time. Simpler

linear projections are up to three orders of magnitudes faster once phrasal representations have been learned and can be as accurate. Our novel local linear projection is much more accurate than the global projection of Mikolov et al. (2013a) at only a small increase in running time. This brings us closer to generating new translation rules on-the-fly for unseen sentences. Finally, we showed that redundant bit vectors are three times faster but also significantly more accurate than locality sensitive hashing in our setting. To our knowledge this is the first application of redundant bit vectors on a natural language processing task.

In future work, we would like to investigate more elaborate projection schemes that use contextual information from the source side or non-linear projections. Furthermore, we would like to apply redundant bit vectors to other NLP tasks.

Acknowledgment

We thank the three anonymous reviewers for helpful suggestions. We are also grateful to Chris Quirk, Kristina Toutanova, Jonathan Clark, Qin Gao, Austin Matthews, Liang Huang, Mingbo Ma, and Mo Yu for discussion. Kai Zhao was partially supported by DARPA FA8750-13-2-0041 (DEFT).

References

Chris Callison-Burch, Philipp Koehn, and Miles Osborne. 2006. Improved statistical machine translation using paraphrases. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 17–24. Association for Computational Linguistics.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.

Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 253–262. ACM.

Michel Galley and Christopher D Manning. 2008. A simple and effective hierarchical phrase reordering model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 848–856. Association for Computational Linguistics.

Jianfeng Gao, Xiaodong He, Wen-tau Yih, and Li Deng. 2014. Learning continuous phrase representations for translation modeling. In *Proc. ACL*.

Jonathan Goldstein, John C Plat, and Christopher JC Burges. 2005. Redundant bit vectors for quickly searching high-dimensional regions. In *Deterministic and Statistical Methods in Machine Learning*, pages 137–158. Springer.

Amit Goyal, Hal Daumé III, and Raul Guerra. 2012. Fast large-scale approximate graph construction for nlp. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1069–1080. Association for Computational Linguistics.

Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *ACL*, volume 2008, pages 771–779. Citeseer.

Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*.

Shujie Liu, Chi-Ho Li, Mu Li, and Ming Zhou. 2012. Learning translation consensus with structured label propagation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 302–310. Association for Computational Linguistics.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their composition-

ality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Tomas Mikolov, Yih Wen-tau, and Zweig Geoffrey. 2013c. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 746–751. Association for Computational Linguistics.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive science*, 34(8):1388–1429.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. 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.

Saša Petrović, Miles Osborne, and Victor Lavrenko. 2010. Streaming first story detection with application to twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 181–189. Association for Computational Linguistics.

Reinhard Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 320–322. Association for Computational Linguistics.

Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. 2005. Randomized algorithms and nlp: using locality sensitive hash function for high speed noun clustering. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 622–629. Association for Computational Linguistics.

Sam T Roweis and Lawrence K Saul. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326.

Avneesh Saluja, Hany Hassan, Kristina Toutanova, and Chris Quirk. 2014. Graph-based semi-supervised learning of translation models from monolingual data. In *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Richard Socher, Eric Huang, Jeffrey Pennington, Andrew Y Ng, and Christopher D. Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems 24*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2014. Word representations: a simple and general method for semi-supervised learning. In *Proc. ACL*.

Jiajun Zhang, Shujie Liu, Mu Li, Ming Zhou, and Chengqing Zong. 2014. Bilingually-constrained phrase embeddings for machine translation. In *Proceedings of the 52th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics*.

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