Bilingual Lexicon Induction by Learning to Combine Word-Level and Character-Level Representations

Geert Heyman[1]**, Ivan Vulić**[2]**, and Marie-Francine Moens**[1]

[1]LIIR, Department of Computer Science, KU Leuven
[2]Language Technology Lab, DTAL, University of Cambridge
{geert.heyman,sien.moens}@cs.kuleuven.be
iv250@cam.ac.uk

Abstract

We study the problem of bilingual lexicon induction (BLI) in a setting where some translation resources are available, but unknown translations are sought for certain, possibly domain-specific terminology. We frame BLI as a classification problem for which we design a neural network based classification architecture composed of recurrent long short-term memory and deep feed forward networks. The results show that word- and character-level representations each improve state-of-the-art results for BLI, and the best results are obtained by exploiting the synergy between these word- and character-level representations in the classification model.

1 Introduction

Bilingual lexicon induction (BLI) is the task of finding words that share a common meaning across different languages. Automatically induced bilingual lexicons support a variety of tasks in information retrieval and natural language processing, including cross-lingual information retrieval (Lavrenko et al., 2002; Levow et al., 2005; Vulić and Moens, 2015; Mitra et al., 2016), statistical machine translation (Och and Ney, 2003; Zou et al., 2013), or cross-lingual entity linking (Tsai and Roth, 2016). In addition, they serve as a natural bridge for cross-lingual annotation and model transfer from resource-rich to resource-impoverished languages, finding their application in downstream tasks such as cross-lingual POS tagging (Yarowsky and Ngai, 2001; Täckström et al., 2013; Zhang et al., 2016), dependency parsing (Zhao et al., 2009; Durrett et al., 2012; Upadhyay et al., 2016), semantic role labeling (Padó and Lapata, 2009; van der Plas et al., 2011), to name only a few.

Current state-of-the-art BLI results are obtained by cross-lingual word embeddings (Mikolov et al., 2013b; Faruqui and Dyer, 2014; Gouws et al., 2015; Vulić and Moens, 2016; Duong et al., 2016, inter alia). They significantly outperform traditional count-based baselines (Gaussier et al., 2004; Tamura et al., 2012). Although cross-lingual word embedding models differ on the basis of a bilingual signal from parallel, comparable or monolingual data used in training (e.g., word, sentence, document alignments, translation pairs from a seed lexicon),[1] they all induce word translations in the same manner. (1) They learn a *shared bilingual semantic space* in which all source language and target language words are represented as dense real-valued vectors. The shared space enables words from both languages to be represented in a uniform language-independent manner such that similar words (regardless of the actual language) have similar representations. (2) Cross-lingual semantic similarity between words w and v is then computed as $SF(\vec{w}, \vec{v})$, where $\vec{w}$ and $\vec{v}$ are word representations in the shared space, and SF denotes a similarity function operating in the space (cosine similarity is typically used). A target language word v with the highest similarity score $\arg\max_v SF(\vec{w}, \vec{v})$ is then taken as the correct translation of a source language word w.

In this work, we detect two major gaps in current representation learning for BLI. First, the standard embedding-based approach to BLI learns representations solely on the basis of word-level information. While early BLI works already established that character-level orthographic features may serve as useful evidence for identifying translations (Melamed, 1995; Koehn and Knight, 2002;

[1] See recent comparative studies on cross-lingual word embedding learning (Upadhyay et al., 2016; Vulić and Korhonen, 2016) for an in-depth discussion of the differences in modeling and bilingual supervision.

Haghighi et al., 2008), there has been no attempt to learn character-level bilingual representations automatically from the data and apply them to improve on the BLI task. Moreover, while prior work typically relies on simple orthographic distance measures such as edit distance (Navarro, 2001), we show that such character-level representations can be induced from the data. Second, Irvine and Callison-Burch (2013; 2016) demonstrated that bilingual lexicon induction may be framed as a classification task where multiple heterogeneous translation clues/features may be easily combined. Yet, all current BLI models still rely on straightforward similarity computations in the shared bilingual word-level semantic space (see Sect. 2).

Motivated by these insights, we propose a *novel bilingual lexicon induction (BLI) model* that combines automatically extracted word-level and character-level representations in a classification framework. As the seminal bilingual representation model of Mikolov et al. (2013b), our bilingual model learns from a set of training translation pairs, but we demonstrate that the synergy between word-level and character-level features combined within a deep neural network based classification framework leads to improved BLI results when evaluated in the medical domain. BLI has a large value in finding translation pairs in specialized domains such as the medical domain, where general translation resources are often insufficient to capture translations of all domain terminology.

This paper has several contributions:

(C1) *On the word level*, we show that framing BLI as a classification problem, that is, using word embeddings as features for classification leads to improved results compared to standard embedding-based BLI approaches (Mikolov et al., 2013b; Vulić and Korhonen, 2016) which rely on similarity metrics in a bilingual semantic space.

(C2) *On the character level*, we find that learning character-level representations with an RNN architecture significantly improves results over standard distance metrics used in previous BLI research to operationalize orthographic similarity.

(C3) We finally show that it is possible to effectively *combine word- and character-level signals* using a deep feed-forward neural network. The combined model outperforms "single" word-level and character-level BLI models which rely on only one set of features.

2 Background

Word-Level Information for BLI Bilingual lexicon induction is traditionally based on word-level features, aiming at quantifying cross-lingual word similarity on the basis of either (1) context vectors, or (2) automatically induced bilingual word representations. A typical context-vector approach (Rapp, 1995; Fung and Yee, 1998; Gaussier et al., 2004; Laroche and Langlais, 2010; Vulić and Moens, 2013b; Kontonatsios et al., 2014, inter alia) constructs context vectors in two languages using weighted co-occurrence patterns with other words, and a bilingual seed dictionary is then used to translate the vectors. Second-order BLI approaches which represent a word by its monolingual semantic similarity with other words were also proposed, e.g., (Koehn and Knight, 2002; Vulić and Moens, 2013a), as well as models relying on latent topic models (Vulić et al., 2011; Liu et al., 2013).

Recently, state-of-the-art BLI results were obtained by a suite of *bilingual word embedding* (BWE) models. Given source and target language vocabularies V^S and V^T, all BWE models learn a representation of each word $w \in V^S \sqcup V^T$ as a real-valued vector: $\vec{w} = [ft_1, \ldots, ft_d]$, where $ft_k \in \mathbb{R}$ denotes the value for the k-th cross-lingual feature for w within a d-dimensional shared bilingual embedding space. Semantic similarity $sim(w, v)$ between two words $w, v \in V^S \sqcup V^T$ is then computed by applying a similarity function (SF), e.g. cosine (*cos*) on their representations in the bilingual space: $sim(w, v) = SF(\vec{w}, \vec{v}) = cos(\vec{w}, \vec{v})$.

A plethora of variant BWE models were proposed, differing mostly in the strength of bilingual supervision used in training (e.g., word, sentence, document alignments, translation pairs) (Zou et al., 2013; Mikolov et al., 2013b; Hermann and Blunsom, 2014; Chandar et al., 2014; Søgaard et al., 2015; Gouws et al., 2015; Coulmance et al., 2015; Vulić and Moens, 2016, inter alia). Although the BLI evaluation of the BWE models was typically performed on Indo-European languages, none of the works attempted to learn character-level representations to enhance the BLI performance.

In this work, we experiment with two BWE models that have demonstrated a strong BLI performance using only a small seed set of word translation pairs (Mikolov et al., 2013b), or document alignments (Vulić and Moens, 2016) for bilingual supervision.

It is also important to note that other word-level translation evidence was investigated in the literature. For instance, the model of Irvine and Callison-Burch (2016) relies on raw word frequencies, temporal word variations, and word burstiness. As the main focus of this work is investigating the combination of automatically induced word-level and character-level representations, we do not exploit the whole space of possible word-level features and leave this for future work. However, we stress that our framework enables the inclusion of these additional word-level signals.

Character-Level Information for BLI For language pairs with common roots such as English-Dutch or English-Spanish, translation pairs often share orthographic character-level features, and regularities (e.g., *ideal:ideaal*, *apparition:aparición*). Orthographic translation clues are even more important in certain domains such as medicine, where words with the same roots (from Greek and Latin), and abbreviations are frequently encountered (e.g., *D-dimer:D-dimeer*, *meiosis:meiose*). When present, such orthographic clues are typically strong indicators of translation pairs (Haghighi et al., 2008). This observation was exploited in BLI, applying simple string distance metrics such as Longest Common Subsequence Ratio (Melamed, 1995; Koehn and Knight, 2002), or edit distance (Mann and Yarowsky, 2001; Haghighi et al., 2008). Irvine and Callison-Burch (2016) showed that these metrics may be used with languages with different scripts: they transliterate all words to the Latin script before calculating normalized edit distance.

BLI as a Classification Task Irvine and Callison-Burch (2016) demonstrate that BLI can be observed as a classification problem. They train a linear classifier to combine similarity scores from different signals (e.g., temporal word variation, normalized edit distance, word burstiness) using a set of training translation pairs. The approach outperforms an unsupervised combination of signals based on a mean reciprocal rank aggregation, as well as the matching canonical correlation analysis algorithm of Haghighi et al. (2008). A drawback of their classification framework is that translation signals are processed (i.e., converted to a similarity score) and weighted independently.

In contrast to their work, we propose to learn character-level representations instead of using the simple edit distance signal between candidate translations. In addition, our model identifies translations by jointly processing and combining character-level and word-level translation signals.

3 Methodology

In this paper we frame BLI as a classification problem as it supports an elegant combination of word-level and character-level representations. Let V^S and V^T denote the sets of all unique source and target words respectively, and C^S and C^T denote the sets of all unique source and target characters. The goal is to learn a function $g : X \to Y$, where the input space X consists of all candidate translation pairs $V^S \times V^T$ and the output space Y is $\{-1, +1\}$. We define g as:

$$g(w^S, w^T) = \begin{cases} +1 & \text{, if } f(w^S, w^T) > t \\ -1 & \text{, otherwise} \end{cases}$$

Here, f is a function realized by a neural network that outputs a classification score between 0 and 1; t is a threshold tuned on a validation set. When the neural network is confident that w^S and w^T are translations, $f(w^S, w^T)$ will be close to 1. The reason for placing a threshold t on the output of f is twofold. First, it allows balancing between recall and precision. Second, the threshold naturally accounts for the fact that words might have multiple translations: if two target language words w^T_1 and w^T_2 both have high scores when paired with w^S, both may be considered translations of w^S.

Since neural network parameters are trained using a set of positive translation pairs D_{lex}, one way to interpret f is to consider it an automatically trained similarity function. For each positive training translation pair $< w^S, w^T >$, we create $2N_s$ *noise* or *negative* training pairs. These negative samples are generated by randomly sampling N_s target language words $w^T_{neg,S,i}$, $i = 1, \ldots, N_s$ from V^T and pairing them with the source language word w^S from the true translation pair $< w^S, w^T >$.[2] Similarly, we randomly sample N_s source language words $w^S_{neg,T,i}$ and pair them with w^T to serve as negative samples. We then train the network by minimizing cross-entropy loss, expressed by Eq. (1):

[2] If we accidentally construct a negative pair which occurs in the set of positive pairs D_{lex}, we re-sample until we obtain exactly N_s negative samples.

$$\mathcal{L}_{ce} = \sum_{<w_s,w_t>\in D_{lex}} \Big(\log(f(w^S, w^T)) - \sum_{i=1}^{N_s} \log(f(w^S_{neg,T,i}, w^T)) - \sum_{i=1}^{N_s} \log(f(w^S, w^T_{neg,S,i}))\Big) \tag{1}$$

We further explain the architecture of the neural network and the strategy we use to identify candidate translations during prediction. Four key components may be distinguished: (1) the input layer; (2) the character-level encoder; (3) the word-level encoder; and (4) a feed-forward network that combines the output representations from the two encoders into the final classification score.

3.1 Input Layer

The goal is to exploit the knowledge encoded in both the word and character levels. Therefore, the raw input representation of a word $w \in V^S$ of character length M consists of (1) its *one-hot* encoding on the word level, labeled x^S_w; and (2) a sequence of M one-hot encoded vectors $x^S_{c0}, .., x^S_{ci}, ..x^S_{cM}$ on the character level, representing the character sequence of the word. x^S_w is thus a $|V^S|$-dimensional word vector with all zero entries except for the dimension that corresponds to the position of the word in the vocabulary. x^S_{ci} is a $|C^S|$-dimensional character vector with all zero entries except for the dimension that corresponds to the position of the character in the character vocabulary C^S.

3.2 Character-Level Encoder

To encode a pair of character sequences $x^S_{c0}, .., x^S_{ci}, ..x^S_{cn}$, $x^T_{c0}, .., x^T_{ci}, ..x^T_{cm}$ we use a two-layer long short-term memory (LSTM) recurrent neural network (RNN) (Hochreiter and Schmidhuber, 1997) as illustrated in Fig. 1. At position i in the sequence, we feed the concatenation of the i^{th} character of the source language and target language word from a training pair to the LSTM network. The characters are represented by their one-hot encoding. To deal with the possible difference in word length, we append special padding characters at the end of the shorter word (see Fig. 1). s_{1i}, and s_{2i} denote the states of the first and second layer of the LSTM. We found that a two-layer LSTM performed better than a shallow LSTM. The output at the final state s_{2N} is the character-level representation r^{ST}_c. We apply dropout regularization (Srivastava et al., 2014) with a keep probability of 0.5 on the output connections of the LSTM (see the dotted lines in Fig. 1). We will further refer to this architecture as CHAR-LSTM$_{joint}$. [3]

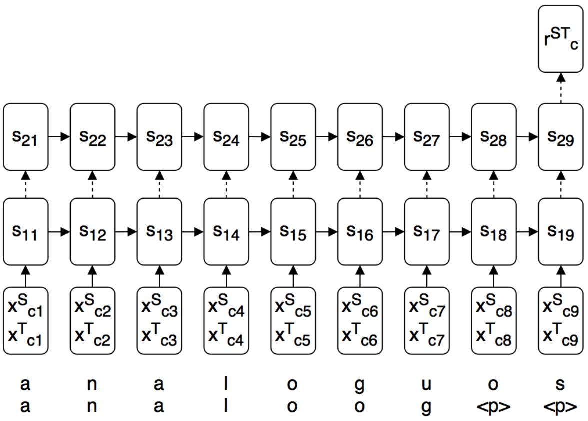

r^{ST}_c
s_{21} s_{22} s_{23} s_{24} s_{25} s_{26} s_{27} s_{28} s_{29}
s_{11} s_{12} s_{13} s_{14} s_{15} s_{16} s_{17} s_{18} s_{19}
x^S_{c1} x^T_{c1} x^S_{c2} x^T_{c2} x^S_{c3} x^T_{c3} x^S_{c4} x^T_{c4} x^S_{c5} x^T_{c5} x^S_{c6} x^T_{c6} x^S_{c7} x^T_{c7} x^S_{c8} x^T_{c8} x^S_{c9} x^T_{c9}
a n a l o g u o s
a n a l o o g <p> <p>

Figure 1: An illustration of the character-level LSTM encoder architecture using the example EN-NL translation pair <*analogous, analoog*>.

3.3 Word-Level Encoder

We define the word-level representation of a word pair $< w^S, w^T >$ simply as the concatenation of word embeddings for w^S and w^T:

$$r^{ST}_w = W^S \cdot x^S_w \quad || \quad W^T \cdot x^T_w \tag{2}$$

Here, r^{ST}_w is the representation of the word pair, and W^S, W^T are word embedding matrices looked up using one-hot vectors x^S_w and x^T_w. The first variant of the architecture assumes that W^S and W^T are obtained in advance using any state-of-the-art word embedding model, e.g., (Mikolov et al., 2013b; Vulić and Moens, 2016). They are then kept *fixed* when minimizing the loss from Eq. (1). In Sect. 5.3, however, we investigate another variant architecture where word embeddings are optimized *jointly* with their unsupervised context-prediction objective and the cross-entropy loss from Eq. (1).

To test the generality of our approach, we experiment with two well-known embedding models: (1) the model from Mikolov et al. (2013b), which trains monolingual embeddings using skip-gram with negative sampling (SGNS) (Mikolov et al., 2013a); and (2) the model of Vulić and Moens (2016) which learns word-level bilingual embeddings from document-aligned comparable

[3] A possible modification to the architecture would be to swap the (unidirectional) LSTM for a bidirectional LSTM. In preliminary experiments on the development set this did not yield improvements over the proposed architecture, we thus do not discuss it further.

data (BWESG). For both models, the top layers of our proposed classification network should learn to relate the word-level features stemming from these word embeddings using a set of annotated translation pairs.

3.4 Combination: Feed-Forward Network

To combine representations on word- and character-level we use a fully connected feed-forward neural network r_h on top of the concatenation of r_w^{ST} and r_c^{ST} which is fed as the input to the network:

$$r_{h_0} = r_w^{ST} \quad \| \quad r_c^{ST} \tag{3}$$

$$r_{h_i} = \sigma(W_{h_i} \cdot r_{h_{i-1}} + b_{h_i}) \tag{4}$$

$$score = \sigma(W_o \cdot r_{h_H} + b_o) \tag{5}$$

σ denotes the sigmoid function and H denotes the number of layers between the representation layer and the output layer. In the simplest architecture, H is set to 0 and the word-pair representation r_{h_0} is directly connected to the output layer (see Fig. 2A). In this setting each dimension from the concatenated representation is weighted independently. This architecture induces undesirable patterns in the combined activation of features, and consequently does not learn generalizable relationships between source and target language inputs. On the word level, for instance, it is obvious that the classifier needs to combine the embeddings of the source and target word to make an informed decision and not merely calculate a weighted sum of them. Therefore, we opt for an architecture with hidden layers instead (see Fig. 2B). Unless stated otherwise, we use two hidden layers, while in Section 5.3 we further analyze the influence of parameter H.

3.5 Candidate Generation

To identify which word pairs are translations, one could enumerate all translation pairs and feed them to the classifier g. The time complexity of this brute-force approach is $O(|V^S| \times |V^T|)$ times the complexity of g. For large vocabularies this can be a prohibitively expensive procedure. Therefore, we have resorted to a heuristic which uses a noisy classifier: it generates $2N_c << |V^T|$ translation candidates for each source language word w^S as follows. It generates (1) the N_c target words closest to w^S measured by edit distance as translations, and (2) N_c target words measured closest to w^S based on the cosine distance between their word-level embeddings in a bilingual space induced by the embedding model of Vulić and Moens (2016).

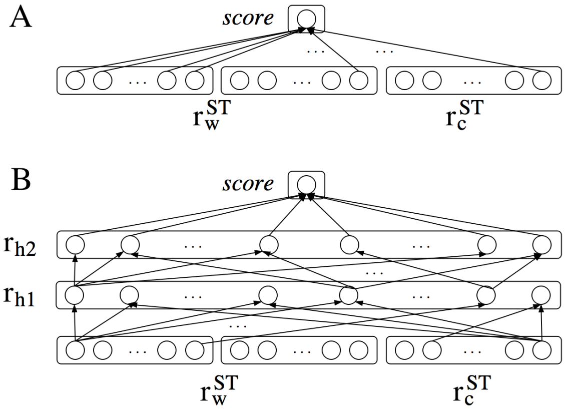


Figure 2: Illustrations of the classification component with feed-forward networks of different depths. A: $H = 0$. B: $H = 2$ (our model). All layers are fully connected.

4 Experimental Setup

Data One of the main advantages of automatic BLI systems is their portability to different languages and domains. However, current standard BLI evaluation protocols still rely on general-domain data and test sets (Mikolov et al., 2013a; Gouws et al., 2015; Lazaridou et al., 2015; Vulić and Moens, 2016, inter alia). To tackle the lack of quality domain-specific data for training and evaluation of BLI models, we have constructed a new English-Dutch (EN-NL) text corpus in the *medical* domain. The corpus contains topic-aligned documents (i.e., for a given document in the source language, we provide a link to a document in the target language that has comparable content). The domain-specific document collection was constructed from the English-Dutch aligned Wikipedia corpus available online[4], where we retain only document pairs with at least 40% of their Wikipedia categories classified as *medical*.[5]

The simple selection heuristic ensures that the main topic of the corpus lies in the medical domain, yielding a final collection of 1198 training document pairs. Following a standard practice (Koehn and Knight, 2002; Haghighi et al., 2008; Prochasson and Fung, 2011), the corpus was then tokenized and lowercased, and words occurring less than five times were filtered out.

Translation Pairs: Training, Development, Test We construct semi-automatically a set of EN-NL translation pairs by translating all words that occur

[4] http://linguatools.org/tools/corpora/

[5] https://www.dropbox.com/s/hlewabraplb9p5n/medicine_en.txt?dl=0

in our pre-processed corpus. This process relied on Google Translate and manual corrections done by fluent EN and NL speakers. Translating the EN vocabulary yields 13,856 translation pairs in total, while the reverse process of translating the NL vocabulary yields 6,537 translation pairs. Taking the union of both lexicons results in 17,567 unique translation pairs, where 7,368 translation pairs (41.94%) have both the source and target language word occurring in our corpus.[6]

We perform a 80/20 random split of the obtained subset of 7,368 translation pairs to construct a training and test set respectively. We make another 80/20 random split of the training set into training and validation data. We note that 20.31% of the source words have more than one translation.

Word-Level Embeddings Skip-gram word embeddings with negative sampling (SGNS) (Mikolov et al., 2013b) are obtained with the word2vec toolkit with the subsampling threshold set to $10e$-4 and window size to 5. BWESG embeddings (Vulić and Moens, 2016) are learned by merging topic-aligned documents with length-ratio shuffling, and then by training a SGNS model over the merged documents with the subsampling threshold set to $10e$-4 and the window size set to 100. The dimensionality of all word-level embeddings is $d = 50$.

Classifier The model is implemented in Python using Tensorflow (Abadi et al., 2015). For training we use the Adam optimizer with default values (Kingma and Ba, 2015) and mini-batches of 10 examples. We used $2N_s = 10$ negative samples and we generated $2N_c = 10$ candidate translation pairs during prediction. The classification threshold t is tuned measuring F_1 scores on the validation set using a grid search in the interval $[0.1, 1]$ in steps of 0.1.

Evaluation Metric The metric we use is F_1, the harmonic mean between recall and precision. While prior work typically proposes only one translation per source word and reports $Accuracy@1$ scores accordingly, here we also account for the fact that words can have multiple translations. We evaluate all models using two different modes: (1) *top* mode, as in prior work, identifies only one translation per source word (i.e., it is the target word with the highest classification score), (2) *all* mode identifies as translation pairs all pairs for which the classification score exceeds threshold t.

5 Results and Discussion

A Roadmap to Experiments We first study automatically extracted word-level and character-level representations and their contribution to BLI in isolation (Sect. 5.1 and Sect. 5.2). It effectively means that for such single-component experiments Eq. 3 is simplified to $r_{h_o} = r_w^{ST}$ (word-level) and $r_{h_o} = r_c^{ST}$ (character-level). Following that, we investigate different ways of combining word-level and character-level representations into improved BLI models (Sect. 5.3). There, we conduct additional analyses which investigate the influence of (i) the number of hidden layers of the classifier, (ii) training data size, and (iii) other variant architectures (i.e., training word-level and character-level representations separately vs. training character-level representations jointly with the classifier vs. training all components jointly).

5.1 Experiment I: Word Level

The goal of this experiment is twofold. First, we want to analyze the potential usefulness of standard word embeddings in a classification framework. Second, we want to compare the BLI approach based on classification to standard BLI approaches that simply compute similarities in a shared bilingual space. All classification NNs are trained for 150 epochs. The results are shown in Tab. 1.

The top two rows are BLI baselines that apply cosine similarity (SIM) in a bilingual embedding space to score translation pairs. For SGNS-based embeddings, we follow (Mikolov et al., 2013b) and align two monolingual embedding spaces by learning a linear mapping using the same set of training translation pairs as used by our classification framework. The BWESG-based embeddings do not exploit available translation pairs, but rely on document alignments during training. The bottom two rows of Tab. 1 use the classification framework we proposed (CLASS).

As the main finding, we see that the classification framework using word-level features outperforms the standard similarity-based framework. BWESG in the similarity-based approach works best in *top*-mode, i.e., it is good at finding a single translation for a source word.[7] The classification-based ap-

[6] Since we use a comparable corpus in our experiments, not all translations of the English vocabulary words occur in the Dutch part of the corpus and vice versa.

[7] Surprisingly, the similarity-based approach with SGNS embeddings (Mikolov et al., 2013b) reports extremely low

	Representation	development F_1 (top)	development F_1 (all)	test F_1 (top)	test F_1 (all)
SIM	BWESG	15.71	11.56	13.43	9.84
	SGNS	0.43	0.40	0.56	0.37
CLASS	BWESG	11.51	16.02	12.12	15.09
	SGNS	**17.67**	**20.67**	**17.25**	**19.79**

Table 1: Comparison of different BLI systems which use only word-level information.

Representation	development F_1 (top)	development F_1 (all)	test F_1 (top)	test F$_1$ (all)
ED$_{norm}$	30.35	31.36	30.89	28.43
log(ED$_{rank}$)	29.01	26.14	29.48	22.25
ED$_{norm}$+ log(ED$_{rank}$)	31.32	30.32	32.27	30.04
CHAR-LSTM$_{joint}$	**33.93**	**35.26**	**33.89**	**34.93**

Table 2: Comparison of character-level BLI methods from prior work with automatically learned character-level representations.

proach is consistently better in translating words with multiple translations as evident from higher all-mode scores in Tab. 1.

When comparing BWESG and SGNS embeddings within the classification framework, we observe that we obtain significantly better results with SGNS embeddings. A plausible explanation is that SGNS embeddings better capture properties related to the local context of a word like syntax information since they are trained with much smaller context windows.[8] We also note that SGNS also has a practical advantage over BWESG concerning the data requirements as the former does not assume document-level alignments.

5.2 Experiment II: Character Level

Here, we compare the representation learned by the character-level encoder with manually extracted features that are commonly used. The following character-level methods are evaluated:

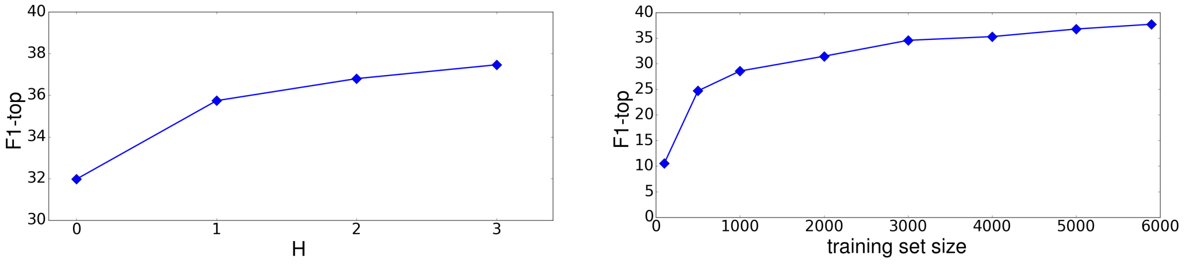
- CHAR-LSTM$_{joint}$, the output of the architecture described in Sect. 3.2
- ED$_{norm}$, the edit distance between the word pair normalized by the average of the number of characters of w_s and w_t as used in prior work (Irvine and Callison-Burch, 2013; Irvine and Callison-Burch, 2016).
- log(ED$_{rank}$), the logarithm of the rank of w_t in a list sorted by the edit distance w.r.t. w_s. This means that the target word that is nearest in edit distance w.r.t. w_s will have a feature value of $log(1) = 0$, words that are more distant from w_s will get higher feature values. Target words with the same edit distance score are assigned the same rank.
- ED$_{norm}$ + log(ED$_{rank}$), the concatenation of the ED$_{norm}$ and log(ED$_{rank}$) features. The combined model results in a stronger baseline.

For the ED-based features we use the same classification framework. However, we use hidden layers only for ED$_{norm}$ + log(ED$_{rank}$) as hidden layers do not make the the one-dimensional feature models (ED$_{norm}$ and log(ED$_{rank}$)) any more expressive. The ED-based models were additionally tuned by performing a grid search to find the optimal values for the number of negative samples $2N_s$ and the number of generated translation candidates $2N_c$. Both $2N_s$ and $2N_c$ are chosen from the interval $[10, 100]$ in steps of 10 based on the performance on the validation set. The ED-based models converge quickly and were only trained for 25 epochs. For the CHAR-LSTM$_{joint}$ representation, we use 512 memory cells per layer, we train the model for 300 epochs, and the parameters $2N_s$ or $2N_c$ were set to the default values (10) without any additional fine-tuning.

The results are displayed in Tab. 2. The overall performance is high compared to the results of the word-level models. The importance of character-level information in this data set is explained by the large amount of medical terminology and expert abbreviations (e.g., *amynoglicosides, aphasics, nystagmus, EPO, EMDR*), which due to its etymological processes, typically contain recurring morphological patterns across languages. It also further supports the need of models that are able to exploit and combine word-level and character-level features. Results also indicate that learning character-level representations from the data is beneficial as the CHAR-LSTM$_{joint}$ model significantly outperforms the baselines used in prior work. The CHAR-LSTM$_{joint}$ shows consistent improvements over baselines across evaluation modes, while the largest gains are again in the all-mode.

results. A possible explanation for such results is that the model is not able to learn a decent linear mapping between two monolingual embedding spaces induced from a small monolingual corpus relying on low-frequency word translation pairs (Vulić and Korhonen, 2016). We verified the influence of low-frequency word pairs by gradually decreasing the amount of pairs in the seed lexicon, keeping only the most frequent word pairs: e.g., limiting the seed lexicon to the 1000 most frequent word pairs, we obtain better results, which are still well below other models in our comparison.

[8] Large window sizes are inherent to the BWESG model.



(a)

(b)

Figure 3: (a) The influence of the number of layers H between the representations and the output layer on the BLI performance; (b) The influence of the training set size (the number of training pairs).

5.3 Experiment III: Combined Model

Encouraged by the excellent results of single-component word-level and character-level BLI models in the classification framework, we also evaluate the combined model. As word-level representations we choose SGNS embeddings and the LSTM consists of 128 in memory cells in each layer in all further experiments.[9] We compare three alternative strategies to learn the parameters of the neural network used for classification:

(1) SEPARATE: Word-level and character-level representations are trained separately. Word-level embeddings and LSTM weights for character-level representations are kept fixed when training the hidden and output layers are simply appended on top of the fixed representations.

(2) CHAR JOINT: Word-level embeddings are trained separately, while character-level representations are trained together with the hidden layers and output layer. This can encourage the network to learn new information on the character-level, different from word-level representations.

(3) ALL JOINT: Motivated by recent work (Ferreira et al., 2016) which proposed a joint formulation for learning task-specific BWEs in a document classification task, all components in our BLI framework are now trained jointly. The joint training objective now consists of two components: the context prediction objective (i.e., SGNS-style objective) and the translation objective described by Eq. (1).

The results are shown in Tab. 3. The CHAR JOINT strategy significantly improves on the best single word-level/character-level models. SEPARATE and ALL JOINT, however, do not improve on the CHAR-LSTM$_{joint}$ model. CHAR JOINT allows the character-level representations to learn features that are complementary to word-level information, which seems crucial for an optimal combination of both representations. Learning word-level embeddings jointly with the rest of the network is not beneficial. This can be explained by the fact that the translation objective deteriorates the generalization capabilities of word embeddings.

	development		test	
training	F_1 (top)	F_1 (all)	F_1 (top)	F_1 (all)
SEPARATE	35.35	35.09	33.60	33.17
CHAR JOINT	**36.78**	**35.85**	**37.73**	**36.61**
ALL JOINT	33.02	33.75	32.86	33.31

Table 3: Results of the combined model (word-level SGNS plus CHAR-LSTM$_{joint}$). Three different strategies of information fusion are compared.

Another crucial parameter is the number of hidden layers H. Fig. 3(a) shows the influence of H on F_1 in *top* mode. BLI performance increases with H. As expected, we see the largest improvement from $H = 0$ to $H = 1$. With $H = 0$ the network is not able to model dependencies between features. More hidden layers allow the network to learn more complex, abstract relations between features, resulting in an improved BLI performance.

Influence of Training Set Size In practice, for various language pairs and domains, one may have at disposal only a limited number of readily available translation pairs. Fig. 3(b) shows the influence of the size of the training set on performance: while it is obvious that more training data leads to a better BLI performance, the results suggest that a competitive BLI performance may be achieved with smaller training sets (e.g., the model reaches up to 77% of the best performance with only $1K$ training pairs, and $> 80\%$ with $2K$ pairs).

6 Conclusion and Future work

We have introduced a neural network based classification architecture for the task of bilingual lexicon

[9] We found that in this setting, where we use both word-level and character-level representations, it is beneficial to use a smaller LSTM than in the character-level only setting.

induction (BLI). We have designed, implemented and evaluated a character-level encoder in the form of a two-layer long short-term memory network and have experimented with different word-level representations. The resulting encodings were used in a deep feed-forward neural network. The results show that especially this combination of character- and word-level knowledge is very successful in the BLI task when evaluated in the medical domain.

Our novel method for learning character-level representations will raise the interest in studying character-level encoders which could be tested in different tasks where string comparisons are important. In future work, we intend to further propose and compare with alternative character-level encoding architectures, and combine additional useful BLI signals in our BLI classification framework.

Acknowledgments

This research has been carried out in the framework of the Smart Computer-Aided Translation Environment (SCATE) project (IWT-SBO 130041). IV is supported by ERC Consolidator Grant LEXICAL (no 648909). The authors are grateful to the anonymous reviewers for their helpful suggestions.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems.

Sarath A.P. Chandar, Stanislas Lauly, Hugo Larochelle, Mitesh M. Khapra, Balaraman Ravindran, Vikas C. Raykar, and Amrita Saha. 2014. An autoencoder approach to learning bilingual word representations. In *Proceedings of NIPS*, pages 1853–1861.

Jocelyn Coulmance, Jean-Marc Marty, Guillaume Wenzek, and Amine Benhalloum. 2015. Trans-gram, fast cross-lingual word embeddings. In *Proceedings of EMNLP*, pages 1109–1113.

Long Duong, Hiroshi Kanayama, Tengfei Ma, Steven Bird, and Trevor Cohn. 2016. Learning crosslingual word embeddings without bilingual corpora. In *Proceedings of EMNLP*, pages 1285–1295.

Greg Durrett, Adam Pauls, and Dan Klein. 2012. Syntactic transfer using a bilingual lexicon. In *Proceedings of EMNLP*, pages 1–11.

Manaal Faruqui and Chris Dyer. 2014. Improving vector space word representations using multilingual correlation. In *Proceedings of EACL*, pages 462–471.

Daniel C. Ferreira, André F. T. Martins, and Mariana S. C. Almeida. 2016. Jointly learning to embed and predict with multiple languages. In *Proceedings of ACL*, pages 2019–2028.

Pascale Fung and Lo Yuen Yee. 1998. An IR approach for translating new words from nonparallel, comparable texts. In *Proceedings of ACL*, pages 414–420.

Éric Gaussier, Jean-Michel Renders, Irina Matveeva, Cyril Goutte, and Hervé Déjean. 2004. A geometric view on bilingual lexicon extraction from comparable corpora. In *Proceedings of ACL*, pages 526–533.

Stephan Gouws, Yoshua Bengio, and Greg Corrado. 2015. BilBOWA: Fast bilingual distributed representations without word alignments. In *Proceedings of ICML*, pages 748–756.

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

Karl Moritz Hermann and Phil Blunsom. 2014. Multilingual models for compositional distributed semantics. In *Proceedings of ACL*, pages 58–68.

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

Ann Irvine and Chris Callison-Burch. 2013. Supervised bilingual lexicon induction with multiple monolingual signals. In *Proceedings of NAACL-HLT*, pages 518–523.

Ann Irvine and Chris Callison-Burch. 2016. A comprehensive analysis of bilingual lexicon induction. *Computational Linguistics*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Philipp Koehn and Kevin Knight. 2002. Learning a translation lexicon from monolingual corpora. In *Proceedings of the ACL Workshop on Unsupervised Lexical Acquisition (ULA)*, pages 9–16.

Georgios Kontonatsios, Ioannis Korkontzelos, Jun'ichi Tsujii, and Sophia Ananiadou. 2014. Combining string and context similarity for bilingual term alignment from comparable corpora. In *Proceedings of EMNLP*, pages 1701–1712.

Audrey Laroche and Philippe Langlais. 2010. Revisiting context-based projection methods for term-translation spotting in comparable corpora. In *Proceedings of COLING*, pages 617–625.

Victor Lavrenko, Martin Choquette, and W. Bruce Croft. 2002. Cross-lingual relevance models. In *Proceedings of SIGIR*, pages 175–182.

Angeliki Lazaridou, Georgiana Dinu, and Marco Baroni. 2015. Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *Proceedings of ACL*, pages 270–280.

Gina-Anne Levow, Douglas W. Oard, and Philip Resnik. 2005. Dictionary-based techniques for cross-language information retrieval. *Information Processing and Management*, 41(3):523–547.

Xiaodong Liu, Kevin Duh, and Yuji Matsumoto. 2013. Topic models+ word alignment= A flexible framework for extracting bilingual dictionary from comparable corpus. In *Proceedings of CoNLL*, pages 212–221.

Gideon S. Mann and David Yarowsky. 2001. Multipath translation lexicon induction via bridge languages. In *Proceedings of NAACL*, pages 1–8.

I. Dan Melamed. 1995. Automatic evaluation and uniform filter cascades for inducing n-best translation lexicons. In *Proceedings of Third Workshop on Very Large Corpora*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Workshop Proceedings of ICLR*.

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013b. Exploiting similarities among languages for machine translation. In *CoRR, abs/1309.4168*.

Bhaskar Mitra, Eric T. Nalisnick, Nick Craswell, and Rich Caruana. 2016. A dual embedding space model for document ranking. *CoRR*, abs/1602.01137.

Gonzalo Navarro. 2001. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Sebastian Padó and Mirella Lapata. 2009. Cross-lingual annotation projection for semantic roles. *Journal of Artificial Intelligence Research*, 36:307–340.

Emmanuel Prochasson and Pascale Fung. 2011. Rare word translation extraction from aligned comparable documents. In *Proceedings of ACL*, pages 1327–1335.

Reinhard Rapp. 1995. Identifying word translations in non-parallel texts. In *Proceedings of ACL*, pages 320–322.

Anders Søgaard, Željko Agić, Héctor Martínez Alonso, Barbara Plank, Bernd Bohnet, and Anders Johannsen. 2015. Inverted indexing for cross-lingual NLP. In *Proceedings of ACL*, pages 1713–1722.

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.

Oscar Täckström, Dipanjan Das, Slav Petrov, Ryan McDonald, and Joakim Nivre. 2013. Token and type constraints for cross-lingual part-of-speech tagging. *Transactions of ACL*, 1:1–12.

Akihiro Tamura, Taro Watanabe, and Eiichiro Sumita. 2012. Bilingual lexicon extraction from comparable corpora using label propagation. In *Proceedings of EMNLP*, pages 24–36.

Chen-Tse Tsai and Dan Roth. 2016. Cross-lingual wikification using multilingual embeddings. In *Proceedings of NAACL-HLT*.

Shyam Upadhyay, Manaal Faruqui, Chris Dyer, and Dan Roth. 2016. Cross-lingual models of word embeddings: An empirical comparison. In *Proceedings of ACL*, pages 1661–1670.

Lonneke van der Plas, Paola Merlo, and James Henderson. 2011. Scaling up automatic cross-lingual semantic role annotation. In *Proceedings of ACL*, pages 299–304.

Ivan Vulić and Anna Korhonen. 2016. On the role of seed lexicons in learning bilingual word embeddings. In *Proceedings of ACL*, pages 247–257.

Ivan Vulić and Marie-Francine Moens. 2013a. Cross-lingual semantic similarity of words as the similarity of their semantic word responses. In *Proceedings of NAACL-HLT*, pages 106–116.

Ivan Vulić and Marie-Francine Moens. 2013b. A study on bootstrapping bilingual vector spaces from non-parallel data (and nothing else). In *Proceedings of EMNLP*, pages 1613–1624.

Ivan Vulić and Marie-Francine Moens. 2015. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *Proceedings of SIGIR*, pages 363–372.

Ivan Vulić and Marie-Francine Moens. 2016. Bilingual distributed word representations from document-aligned comparable data. *Journal of Artificial Intelligence Research*, 55:953–994.

Ivan Vulić, Wim De Smet, and Marie-Francine Moens. 2011. Identifying word translations from comparable corpora using latent topic models. In *Proceedings of ACL*, pages 479–484.

David Yarowsky and Grace Ngai. 2001. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora. In *Proceedings of NAACL*, pages 200–207.

Yuan Zhang, David Gaddy, Regina Barzilay, and Tommi Jaakkola. 2016. Ten pairs to tag - Multilingual POS tagging via coarse mapping between embeddings. In *Proceedings of NAACL-HLT*, pages 1307–1317.

Hai Zhao, Yan Song, Chunyu Kit, and Guodong Zhou. 2009. Cross language dependency parsing using a bilingual lexicon. In *Proceedings of ACL*, pages 55–63.

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