The Importance of Subword Embeddings in Sentence Pair Modeling

Wuwei Lan and Wei Xu
Department of Computer Science and Engineering
Ohio State University
{lan.105, xu.1265}@osu.edu

Abstract
Sentence pair modeling is critical for many NLP tasks, such as paraphrase identification, semantic textual similarity, and natural language inference. Most state-of-the-art neural models for these tasks rely on pretrained word embedding and compose sentence-level semantics in varied ways; however, few works have attempted to verify whether we really need pretrained embeddings in these tasks. In this paper, we study how effective subword-level (character and character n-gram) representations are in sentence pair modeling. Though it is well-known that subword models are effective in tasks with single sentence input, including language modeling and machine translation, they have not been systematically studied in sentence pair modeling tasks where the semantic and string similarities between texts matter. Our experiments show that subword models without any pretrained word embedding can achieve new state-of-the-art results on two social media datasets and competitive results on news data for paraphrase identification.

1 Introduction
Recently, there have been various neural network models proposed for sentence pair modeling tasks, including semantic similarity (Agirre et al., 2015), paraphrase identification (Dolan et al., 2004; Xu et al., 2015), natural language inference (Bowman et al., 2015), etc. Most, if not all, of these state-of-the-art neural models (Yin et al., 2016; Parikh et al., 2016; He and Lin, 2016; Tomar et al., 2017; Shen et al., 2017) have achieved the best performances for these tasks by using pretrained word embeddings, but results without pretraining are less frequently reported or noted. In fact, we will show that, even with fixed randomized word vectors, the pairwise word interaction model (He and Lin, 2016) based on contextual word vector similarities can still achieve strong performance by capturing identical words and similar surface context features. Moreover, pretrained word embeddings generally have poor coverage in social media domain where out-of-vocabulary rate often reaches over 20% (Baldwin et al., 2013).

We investigated the effectiveness of subword units, such as characters and character n-grams, in place of words for vector representations in sentence pair modeling. Though it is well-known that subword representations are effective to model out-of-vocabulary words in many NLP tasks with a single sentence input, such as machine translation (Luong et al., 2015; Costa-jussà and Fonollosa, 2016), language modeling (Ling et al., 2015; Vania and Lopez, 2017), and sequence labeling (dos Santos and Guimarães, 2015; Plank et al., 2016), they are not systematically studied in the tasks that concern pairs of sentences. Unlike in modeling individual sentences, subword representations have impacts not only on the out-of-vocabulary words but also more directly on the relation between two sentences, which is calculated based on vector similarities in many sentence pair modeling approaches (more details in Section 2.1). For example, while subwords may capture useful string similarities between a pair of sentences (e.g. spelling or morphological variations: sister and sista, teach and teaches), they could introduce errors (e.g. similarly spelled words with completely different meanings: ware and war).

To better understand the role of subword embedding in sentence pair modeling, we performed experimental comparisons that vary (1) the type of subword unit, (2) the composition function, and (3) the datasets of different characteristics. We also presented experiments with language modeling as an auxiliary multi-task learning objective, showing consistent improvements. Taken together, subword and language modeling establish
new state-of-the-art results in two social media datasets and competitive results in a news dataset for paraphrase identification without using any pretrained word embeddings.

2 Sentence Pair Modeling with Subwords

The current neural networks for sentence pair modeling (Yin et al., 2016; Parikh et al., 2016; He and Lin, 2016; Liu et al., 2016; Tomar et al., 2017; Wang et al., 2017; Shen et al., 2017, etc) follow a more or less similar design with three main components: (a) contextualized word vectors generated via Bi-LSTM, CNN, or attention, as inputs; (b) soft or hard word alignment and interactions across sentences; (c) and the output classification layer. Different models vary in implementation details, and most importantly, to capture the same essential intuition in the word alignment (also encoded with contextual information) – the semantic relation between two sentences depends largely on the relations of aligned chunks (Agirre et al., 2016). In this paper, we used pairwise word interaction model (He and Lin, 2016) as a representative example and starting point, which reported interaction model as a representation of sentence pairs modeling tasks and the best results by neural models on social media data (Lan et al., 2017).

2.1 Pairwise Word Interaction (PWI) Model

Let \(w^a = (w_1^a, ..., w_m^a)\) and \(w^b = (w_1^b, ..., w_n^b)\) be the input sentence pair consisting of \(m\) and \(n\) tokens, respectively. Each word vector \(w_i \in \mathbb{R}^d\) is initialized with pretrained \(d\)-dimensional word embedding (Pennington et al.; Wieting et al., 2015, 2016), then encoded with word context and sequence order through bidirectional LSTMs:

\[
\h_i^+ = LSTM^b(w_i, \h_{i-1}^+), \\
\h_i^− = LSTM^f(w_i, \h_{i+1}^−), \\
\h_i = [\h_i^+, \h_i^−], \\
h_i = \h_i^+ + \h_i^−.
\]

where \(\h_i^+\) represents forward hidden state, \(\h_i^−\) represents backward hidden state, and \(\h_i^+\) and \(\h_i^−\) are the concatenation and summation of two directional hidden states.

For all word pairs \((w_i^a, w_j^b)\) across sentences, the model directly calculates word pair interactions using cosine similarity, Euclidean distance, and dot product over the outputs of the encoding layer:

\[
D(\h_i^+, \h_j^+) = \cos(\h_i, \h_j), \\
L2Euclid(\h_i^+, \h_j^+), \\
DotProduct(\h_i^+, \h_j^+).
\]

The above equation can also apply to other states \(\h_i^+\) and \(\h_j^+\), resulting in a tensor \(D \in \mathbb{R}^{m \times n}\) after padding one extra bias term. A “hard” attention is applied to the interaction tensor to further enforce the word alignment, by sorting the interaction values and selecting top ranked word pairs. A 19-layer deep CNN is followed to aggregate the word interaction features and the softmax layer to predicate classification probabilities.

2.2 Embedding Subwords in PWI Model

Our subword models only involve modification of the input representation layer in the pairwise interaction model. Let \(c_1, ..., c_k\) be the subword (character unigram, bigram and trigram) sequence of a word \(w\). The subword embedding matrix is \(C \in \mathbb{R}^{d' \times k}\), where each subword is encoded into the \(d'\)-dimensional vector. The same subwords will share the same embeddings. We considered two different composition functions to assemble subword embeddings into word embedding:

Char C2W (Ling et al., 2015) applies Bi-LSTM to subword sequence \(c_1, ..., c_k\), then the last hidden state \(\h_{k, char}^\text{char}\) in forward direction and the first hidden state \(\h_0^\text{char}\) of the backward direction are linearly combined into word-level embedding \(w\):

\[
w = W_f \cdot \h_{k, char}^\text{char} + W_b \cdot \h_0^\text{char} + b \tag{6}
\]

where \(W_f, W_b\) and \(b\) are parameters.

Char CNN (Kim et al., 2016) applies a convolution operation between subword sequence matrix \(C\) and a filter \(F \in \mathbb{R}^{d' \times l}\) of width \(l\) to obtain a feature map \(f \in \mathbb{R}^{k \times l+1}\):

\[
f_j = tanh\left((C \ast [j \text{ : } j + l - 1], F) + b\right) \tag{7}
\]

where \(\langle A, B \rangle = Tr(AB^T)\) is the Frobenius inner product, \(b\) is a bias and \(f_j\) is the \(j\)th element of \(f\). We then take the max-over-time operation to select the most important element:

\[
y_t = \max_j f_j. \tag{8}
\]

After applying \(q\) filters with varied lengths, we can get the array \(w = [y_1, ..., y_q]\), which is followed by a one-layer highway network to generate final word embedding.
Table 1: Statistics of three benchmark datasets for paraphrase identification. The training and testing sizes are in numbers of sentence pairs. The number of unique in-vocabulary (INV) and out-of-vocabulary (OOV) words are calculated based on the publicly available GloVe embeddings (details in Section 3.2).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Training Size</th>
<th>Test Size</th>
<th># INV</th>
<th># OOV</th>
<th>OOV Ratio</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIT-2015</td>
<td>11530</td>
<td>838</td>
<td>7771</td>
<td>1238</td>
<td>13.7%</td>
<td>Twitter trends</td>
</tr>
<tr>
<td>Twitter-URL</td>
<td>42200</td>
<td>9324</td>
<td>24905</td>
<td>11440</td>
<td>31.5%</td>
<td>Twitter/news</td>
</tr>
<tr>
<td>MSRP</td>
<td>4076</td>
<td>1725</td>
<td>16226</td>
<td>1614</td>
<td>9.0%</td>
<td>news</td>
</tr>
</tbody>
</table>

2.3 Auxiliary Language Modeling (LM)

We adapted a multi-task structure, originally proposed by (Rei, 2017) for sequential tagging, to further improve the subword representations in sentence pair modeling. In addition to training the model for sentence pair tasks, we used a secondary language modeling objective that predicts the next word and previous word using softmax over the hidden states of Bi-LSTM as follows:

$$\bar{E}_{LM} = - \sum_{t=1}^{T-1} \log(P(w_{t+1} | \hat{m}_t))$$  \hspace{1cm} (9)

$$\hat{E}_{LM} = - \sum_{t=2}^{T} \log(P(w_{t-1} | \tilde{m}_t))$$  \hspace{1cm} (10)

where $\hat{m}_t = \tanh(W_{hm}\tilde{h}_t)$ and $\tilde{m}_t = \tanh(W_{hm}\tilde{h}_t)$. The Bi-LSTM here is separate from the one in PWI model. The language modeling objective can be combined into sentence pair modeling through a joint objective function:

$$E_{joint} = E + \gamma(\bar{E}_{LM} + \hat{E}_{LM}),$$  \hspace{1cm} (11)

which balances subword-based sentence pair modeling objective $E$ and language modeling with a weighting coefficient $\gamma$.

3 Experiments

3.1 Datasets

We performed experiments on three benchmark datasets for paraphrase identification; each contained pairs of naturally occurring sentences manually labeled as paraphrases and non-paraphrases for binary classification: Twitter URL (Lan et al., 2017) was collected from tweets sharing the same URL with major news outlets such as @CNN. This dataset keeps a balance between formal and informal language. PIT-2015 (Xu et al., 2014, 2015) comes from the Task 1 of Semeval 2015 and was collected from tweets under the same trending topic, which contains varied topics and language styles. MSRP (Dolan and Brockett, 2005) was derived from clustered news articles reporting the same event in formal language. Table 1 shows vital statistics for all three datasets.

3.2 Settings

To compare models fairly without implementation variations, we reimplemented all models into a single PyTorch framework. We followed the setups in (He and Lin, 2016) and (Lan et al., 2017) for the pairwise word interaction model, and used the 200-dimensional GloVe word vectors (Pennington et al.), trained on 27 billion words from Twitter (vocabulary size of 1.2 million words) for social media datasets, and 300-dimensional GloVe vectors, trained on 840 billion words (vocabulary size of 2.2 million words) from Common Crawl for the MSRP dataset. For cases without pretraining, the word/subword vectors were initialized with random samples drawn uniformly from the range [0.05, 0.05]. We used the same hyperparameters in the C2W (Ling et al., 2015) and CNN-based (Kim et al., 2016) compositions for subword models, except that the composed word embeddings were set to 200- or 300- dimensions as the pre-trained word embeddings to make experiment results more comparable. For each experiment, we reported results with 20 epochs.

3.3 Results

Table 2 shows the experiment results on three datasets. We reported maximum F1 scores of any point on the precision-recall curve (Lipton et al., 2014) following previous work.

**Word Models** The word-level pairwise interaction models, even without pretraining (randomized) or fine-tuning (fixed), showed strong performance across all three datasets. This reflects the effective design of the BiLSTM and word interaction layers, as well as the unique character of sentence pair modeling, where n-gram overlapping positively signifies the extent of semantic similarity.
Table 2: Results in F1 scores on Twitter-URL, PIT-2015 and MSRP datasets. The best performance figure in each dataset is denoted in **bold** typeface and the second best is denoted by an *underline*. Without using any pretrained word embeddings, the Subword+LM models achieve better or competitive performance compared to word models.

**3.4 Combining Word and Subword Representations**

In addition, we experimented with combining the pretrained word embeddings and subword models with various strategies: concatenation, weighted average, adaptive models (Miyamoto and Cho, 2016) and attention models (Rei et al., 2016). The weighted average outperformed all others but only showed slight improvement over word-based models in social media datasets; other combination strategies could even lower the performance. The best performance was 0.763 F1 in Twitter-URL and 0.671 in PIT-2015 with a weighted average: $0.75 \times \text{word embedding} + 0.25 \times \text{subword embedding}$.

**4 Conclusion**

We presented a focused study on the effectiveness of subword models in sentence pair modeling and showed competitive results without using pretrained word embeddings. We also showed that subword models can benefit from multi-task language modeling (Subword+LM), the performance on all datasets are further improved, outperforming the best previously reported results by neural models (Lan et al., 2017). A qualitative analysis we conducted reveals that subwords are crucial for out-of-vocabulary words while language modeling ensures more semantic and syntactic compatibility (Table 3).

Table 3: Nearest neighbors of word vectors under cosine similarity in Twitter-URL dataset.
ations are profound. The results shed lights on future work on language-independent paraphrase identification and multilingual paraphrase acquisition, where pretrained word embeddings on large corpora are not readily available in many languages.

Acknowledgments

We thank John Wieting for valuable discussions, and Ohio Supercomputer Center (Center, 2012) for computing resources. Funding was provided by the U.S. Army Research Office (ARO) and Defense Advanced Research Projects Agency (DARPA) under Contract Number W911NF-17-C-0095. The content of the information in this document does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References


