Efficient Purely Convolutional Text Encoding

Szymon Malik*, **Adrian Lancucki***, **Jan Chorowski**
Institute of Computer Science
University of Wrocław
szymon.w.malik@gmail.com, {alan,jch}@cs.uni.wroc.pl

Abstract

In this work, we focus on a lightweight convolutional architecture that creates fixed-size vector embeddings of sentences. Such representations are useful for building NLP systems, including conversational agents. Our work derives from a recently proposed recursive convolutional architecture for auto-encoding text paragraphs at byte level. We propose alternations that significantly reduce training time, the number of parameters, and improve auto-encoding accuracy. Finally, we evaluate the representations created by our model on tasks from *SentEval* benchmark suite, and show that it can serve as a better, yet fairly low-resource alternative to popular bag-of-words embeddings.

1 Introduction

Modern conversational agents often make use of retrieval-based response generation modules [Ram *et al.*, 2018], in which the response of the agent is retrieved from a curated database. The retrieval can be implemented as similarity matching in a vector space, in which natural language sentences are represented as fixed-size vectors. Cosine and Euclidean distances typically serve as similarity measures. Such approaches have been applied by participants of recent chatbot contests: The 2017 Alexa Prize [Pichl *et al.*, 2018; Liu *et al.*, 2017; Serban *et al.*, 2017], and The 2017 NIPS Conversational Intelligence Challenge [Chorowski *et al.*, 2018; Yusupov and Kuratov, 2017]. Retrieval-based modules are fast and predictable. Most importantly, they enable *soft* matching between representations. Apart from this straightforward application in dialogue systems, sentence embeddings are applicable in downstream NLP tasks relevant to dialogue systems. Those include sentiment analysis [Pang and Lee, 2008], question answering [Weissenborn *et al.*, 2017], censorship [Chorowski *et al.*, 2018], or intent detection.

Due to the temporal nature of natural languages, recurrent neural networks gained popularity in NLP tasks. Active research of different architectures led to great advances and eventually a shift towards methods using the transformer architecture [Vaswani *et al.*, 2017] or convolutional layers [Bai *et al.*, 2018; van den Oord *et al.*, 2017a; van den Oord *et al.*, 2017b] among researchers and practitioners alike. In this work, we focus on a lightweight convolutional architecture that creates fixed-size representations of sentences.

Convolutional neural networks have the inherent ability to detect local structures in the data. In the context of conversational systems, their speed and memory efficiency eases deployment on mobile devices, allowing fast response retrieval and better user experience. We analyze and build on the recently proposed Byte-Level Recursive Convolutional Auto-Encoder (BRCA) for text paragraphs [Zhang and LeCun, 2018], which is able to auto-encode text paragraphs into fixed-size vectors, reading in bytes with no additional preprocessing.

Based on our analysis, we are able to explain the behavior of this model, point out possible enhancements, achieve auto-encoding accuracy improvements and an order of magnitude training speed-up, while cutting down the number of parameters by over 70%. We introduce a balanced padding scheme for input sequences and show that it significantly improves convergence and capacity of the model. As we find byte-level encoding unsuitable for embedding sentences, we demonstrate its applicability in processing sentences at word-level. We train the encoder with supervision on Stanford Natural Language Inference corpus [Bowman *et al.*, 2015; Conneau *et al.*, 2017] and investigate its performance on various transfer tasks to assess quality of produced embeddings.

The paper is structured as follows: in Section 2 we introduce some of the notions that appear in the paper. Section 3 discusses relevant work for sentence vector representations. Details of the architecture can be found in Section 4. Section 5 presents the analysis of the auto-encoder and motivations for our improvements. Section 6 demonstrates supervised training of word-level sentence encoder, and evaluates it on tasks relevant to conversational systems. Section 7 concludes the paper.

2 Preliminaries

An information retrieval conversational agent selects a response from a fixed set. Let $D = \{(i_k, r_k)\}_k$ be a set of conversational input-response pairs, and q be a current user's input. Two simple ways of retrieving a response r_k from available data are [Ritter *et al.*, 2011; Chorowski *et al.*, 2018]:

*Equal contribution

- return r_k most similar to user's input q,
- return r_k for which i_k is most similar to q.

Utterances i_k, r_k, q may be represented (embedded) as real-valued vectors.

Many NLP systems represent words as points in a continuous vector space using word embedding methods [Mikolov *et al.*, 2013; Pennington *et al.*, 2014; Bojanowski *et al.*, 2016]. They are calculated based on co-occurrence of words in large corpora. The same methods were applied to obtain sentence embeddings only with partial success, due to the combinatorial explosion of all possible word combinations, which make up a sentence. Instead, Recurrent Neural Networks (RNNs), autoregressive models that can process input sequences of an arbitrary length, are thought to be a good method for handling variable-length textual data, with Long Short-Term Memory network (LSTM) [Hochreiter and Schmidhuber, 1997] being the prime example of such.

Recently, RNNs have been reported to be successfully replaced by convolutional architectures [Bai *et al.*, 2018; van den Oord *et al.*, 2017a; van den Oord *et al.*, 2017b]. Convolutional neural networks are traditionally associated with computer vision and image processing [Krizhevsky *et al.*, 2012; Redmon *et al.*, 2015]. They primarily consist of convolutional layers that apply multiple convolutions to the input, followed by pooling layers that are used for reducing the dimensionality of the hidden state. Convolutional networks are efficient during training and inference: they utilize few parameters and do not require sequential computations, making hardware parallelism easy to use. Due to their popularity in image processing, there are efficient implementations that scale well.

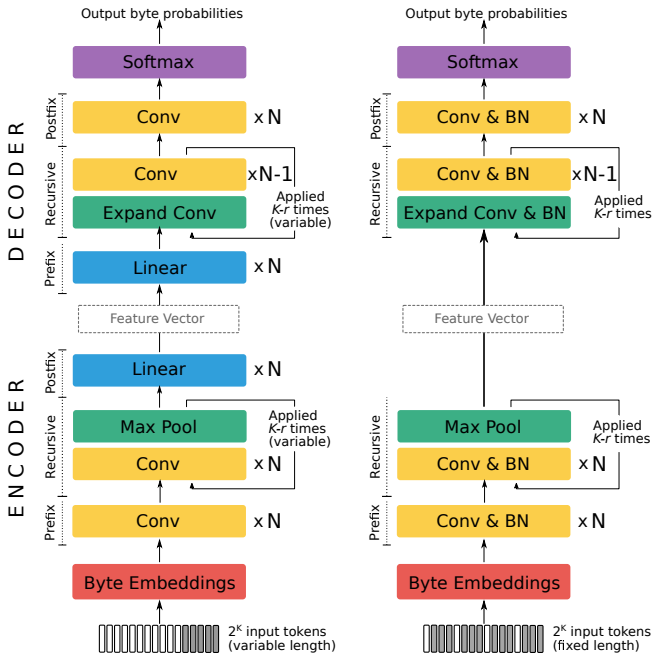
Residual connection [He *et al.*, 2015] is a connection that adds an unchanged input to the output of the layer or block of layers. During the forward pass it provides upper layers with undistorted signal from the input and intermediate layers. During the backward pass it mitigates vanishing and exploding gradient problems [Hochreiter *et al.*, 2001].

Batch Normalization [Ioffe and Szegedy, 2015] (BN) applies normalization to all activations in every minibatch. Typical operations used in neural networks are sensitive to changes in the range and magnitude of inputs. During training the inputs to upper layers vary greatly due to changes in weights of the model. Normalization of the signal in each layer has the potential to alleviate this problem. Both BN and residual connections enable faster convergence by helping with forward and backward flow of information. They were also crucial in training our models.

3 Related Work

There are many methods for creating sentence embeddings, the simplest being averaging word-embedding vectors of a given sentence [Joulin *et al.*, 2016; Le and Mikolov, 2014]. SkipThought [Kiros *et al.*, 2015] generalizes idea of unsupervised learning of *word2vec* word embeddings [Mikolov *et al.*, 2013]. It is implemented in the encoder-decoder setting using LSTM networks. Given a triplet of consecutive sentences (s_{i-1}, s_i, s_{i+1}), the encoder creates a fixed-size embedding vector of the sentence s_i, and the decoder tries to generate sentences s_{i-1} and s_{i+1} from this representation. SkipThought vectors have been shown to preserve syntactic and semantic properties [Kiros *et al.*, 2015], so that their similarity is represented in the embedding space.

InferSent model [Conneau *et al.*, 2017] shows that training embedding systems with supervision on a natural language inference task may be superior to an unsupervised training. Recently, better results were obtained by combining supervised learning on an auxiliary natural language inference corpus with learning to predict the correct response on a conversational data corpus [Yang *et al.*, 2018].



(a) BRCA (b) Our model

Figure 1: Structural comparison of Byte-Level Recursive Convolutional Auto-Encoder (BRCA) and our model. Dark boxes indicate input padding. BRCA pads the input form the right to the nearest power of two. We pad the input evenly to a fixed-size vector. Our model does not have postfix/prefix groups with linear layers and uses Batch Normalization (BN) after every layer.

4 Model Description

Our model builds on the Byte-Level Recursive Convolutional Auto-Encoder [Zhang and LeCun, 2018] (BRCA). Both models use a symmetrical encoder and a decoder. They encode a variable-length input sequence as a fixed-size latent representation, by applying a variable number of convolve-and-pool stages (Figure 1). Unlike in autoregressive models, the recursion is not applied over the input sequence length, but over the depth of the model. As a result, each recursive step processes all sequence elements in parallel.

4.1 Encoder

Our encoder uses two operation groups from the BRCA: a prefix group and a recursive group. The **prefix group** consists of N temporal convolutional layers, and the **recursive group**

consists of N temporal convolutional layers followed by a max-pool layer with kernel size 2. For each sequence, the prefix group is applied only once, while the recursive group is applied multiple times, sharing weights between all applications. All convolutional layers have $d = 256$ channels, kernels of size 3 and are organized into residual blocks, with 2 layers per block, ReLU activations, residual connections, and Batch Normalization (see Section 5.5 for details).

The encoder of our model reads in text, by sentence or by paragraph, as a sequence of discrete tokens (e.g. bytes, characters, words). Each input token is embedded as a fixed-length d-dimensional vector. Unlike [Zhang and LeCun, 2018], where input sequence is zero-padded to the nearest power of 2, we pad the input to a fixed length 2^K, distributing the characters evenly across the input. We motivate this decision by the finding that the model zero-padded to the nearest power of 2 does not generalize to sentences longer than those seen in the training data (see Section 5.2).

First, the prefix group is applied, which retains the dimensionality and the number of channels, i.e., sequence length and embedding size. Let $d \cdot 2^r$ be the dimensionality of the latent code output by the encoder. The encoder then applies the recursive group $K - r$ times. Note that with r one may control size of a latent vector (the level of the compression). With the max-pooling, every application halves the length of the sequence. Weights are shared between applications. Finally, the encoder outputs a latent code of size $d \cdot 2^r$. Unlike [Zhang and LeCun, 2018], we do not apply any linear layers after recursions. Our experiments have shown that they slightly degrade the performance of the model, and constitute the majority of its parameters.

4.2 Decoder

The decoder acts in reverse. First, it applies a **recursive group** consisting of a convolutional layer which doubles the number of channels to $2d$, which is followed by an expand transformation [Zhang and LeCun, 2018] and $N - 1$ convolutional layers. Then it applies a **postfix group** of N temporal convolutional layers. Similarly to the encoder, the layers are organized into residual blocks with ReLU activations, Batch Normalization, and have the same dimensionality and kernel size. We double the size of input in the residual connection, which bypasses the first two convolutions of the recursive group, by stacking it with itself. We found it crucial for convergence speed to use only residual blocks in the network, also in the expand block.

The decoder applies its recursive group $K - r$ times. Each application doubles the numbers of channels, while the expand transformation reorders and reshapes the data to effectively double the length of the sequence while the number of channels is unchanged and equals d. The postfix group processes a tensor of size $2^K \times d$ and retains its dimensionality. The output is interpreted as 2^K probability distributions over possible output elements. Adding an output embedding layer, either tied with input embedding layer or separate, slowed down training and did not improve the results. At the end, a Softmax layer is used to compute output probabilities over possible bytes. Note that output probabilities are independent from one another conditioned on the input.

5 Model Analysis

In this section we justify our design choices through a series of experiments with the BRCA model and report their outcomes.[1]

5.1 Data

In order to produce comparable results, we prepared an English Wikipedia dataset with similar sentence length distribution to [Zhang and LeCun, 2018]. Namely, we took at random 11 million sentences from an English Wikipedia dump extracted with WikiExtractor[2], so that their length distribution would roughly match that of [Zhang and LeCun, 2018] (Table 1). In experiments with random data, we generate random strings of `a-zA-Z0-9` ASCII characters.

Table 1: Lengths of paragraphs in the English Wikipedia dataset

Length	Percentage
4-63 B	35%
64-127 B	14%
128-255 B	20%
256-511 B	18%
512-1023 B	14%

5.2 Model Capacity

Natural language texts are highly compressible due to their low entropy, which results from redundancy of the language [Levitin and Reingold, 1994]. In spite of this, the considered models struggle to auto-encode 1024-byte short paragraphs into 1024-float latent vectors, which are 4096-byte given their sheer information content. Transition from discrete to continuous representation and inherent inefficiency of the model are likely to account for some of this overhead.

One can imagine an initialization of weights that, given the over-capacitated latent representation, would make the network perform identity for paragraphs up to 128 bytes long[3]. We confirmed those speculations experimentally, training models on paragraphs of random printable ASCII characters, namely random strings of `a-zA-Z0-9` symbols (Table 2). The empirical capacity of our model is 128 bytes, which sheds light on the amount of overhead. This model has to be trained on paragraphs longer than 512 bytes in order to learn useful, compressing behavior given a 1024-float latent representation.

5.3 Generalization to Longer Sequences

Auto-encoding RNN models such as LSTM are known to deteriorate gradually with longer sequences [Cho *et al.*, 2014; Chorowski *et al.*, 2015]. We trained a BRCA model ($N = 2$) and a LSTM encoder-decoder network with hidden size 256.

[1] Source code of our models is available: `https://github.com/smalik169/recursive-convolutional-autoencoder`

[2] `https://github.com/attardi/wikiextractor`

[3] When max-pooling is replaced by convolution with stride 2 and kernel size 2

Table 2: Learning identity by training on random sequences of ASCII characters of different length. Accuracy is presented for BRCA (N=8) model.

Training Lengths	Test Length	Accuracy
4 – 128	128	99.81%
4 – 512	128	60.79%
	256	22.99%
	512	9.81%

Table 3: Comparison of the ability of BRCA and LSTM encoder-decoder to learn an identity function and generalize to unseen data. Values represent byte-level decoding accuracy. Note that the LSTM decoder has the advantage of always being primed with the correct prefix sequence.

Lengths (bytes)	BRCA (N=2)	LSTM-LSTM
9-16	97.06%	91.17%
17-32	97.96%	90.20%
33-64	97.45%	91.72%
65-128	83.56%	86.34%
129-256	11.66%	72.88%
257-512	8.05%	58.80%

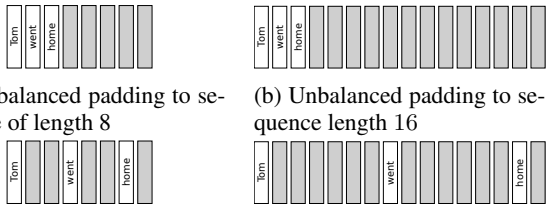
Both models were trained on sentences of length up to 128 bytes and evaluated on unseen data. The LSTM model did not perfectly learn the identity function, even though it was solving an easier task of predicting the character given the correct prefix. However, the LSTM model generalized much better on longer sequences, where performance of BRCA deteriorated rapidly (Table 3).

5.4 Balanced Padding of Input Sequences

We found BRCA difficult to train. The default hyperparameters given by the authors [Zhang and LeCun, 2018] are single-sample batches, SGD with momentum 0.9, and a small learning rate 0.01 with 100 epochs of training. In our preliminary experiments, increasing the batch size by batching paragraphs of the same length improved convergence on datasets with short sentences (mostly up to 256 bytes long), but otherwise deteriorated on the Wikipedia dataset, where roughly 50% paragraphs are longer than 256 bytes. We suspect that the difficulty lies in the difference of the underlying tasks: long paragraphs require compressive behavior, while short ones merely require learning the identity function. Updating network parameters towards one tasks hinders the performance on the others, hence the necessity for careful training.

In order to blend in both tasks, we opted for padding input sequences into fixed-length vectors. We find it sensible to fix maximum length of input sentence, since the model does not generalize to unseen lengths anyway. Variable length of input in BRCA does save computations, however we found fixing input size to greatly improve training time, despite the overhead.

In order to make the tasks more similar, we propose balanced padding of the inputs (Figure 2). Instead of padding from the right up to 2^K bytes, we pad to the nearest power of 2 and distribute the remaining padding equally in between the bytes. We hypothesized that it could free convolutional layers from the burden of propagating the signal from left to right in order to fill the whole latent vector, as it would be the case, e.g., when processing a 64-byte paragraph padded with 960 empty tokens from the right to form 1024-byte input. Empirically, this trades additional computations for better convergence characteristics.



(a) Unbalanced padding to sequence of length 8

(b) Unbalanced padding to sequence length 16

(c) Balanced padding to sequence of length 8

(d) Balanced padding to sequence length 16

Figure 2: Unbalanced and balanced padding of an input sequence to a fixed-length sequence. Grey boxes are (zero) padding, white boxes are input embedding vectors.

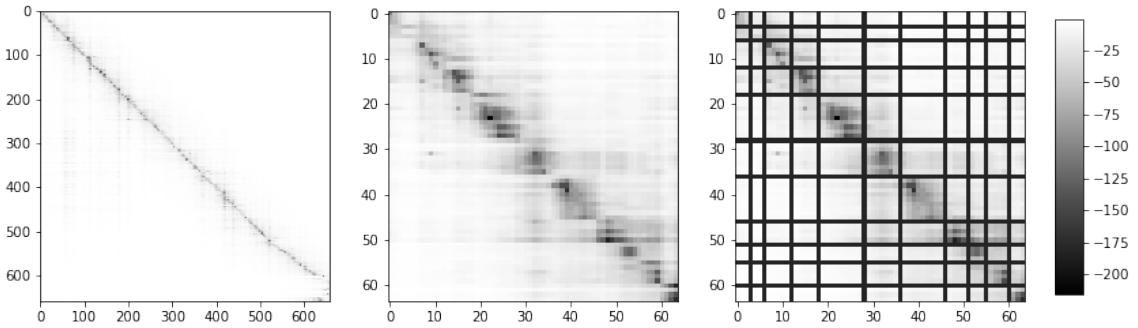
5.5 Batch Normalization

Fixed-length, balance-padded inputs allow easy mixing of paragraphs of different lengths within a batch, in consequence allowing raising the batch size, applying Batch Normalization and raising the learning rate. This enables a significant speed-up in convergence and better auto-encoding accuracy (see Section 5.6). However, the statistics collected by BN layers differ during each of the $K - r$ recursive steps, even though the weights of convolutions in the recursive layers are shared. This breaks auto-encoding during inference time, when BN layers have fixed mean and standard deviation collected over a large dataset. We propose to alleviate this issue by either: a) collecting separate statistics for each recursive application and each input length separately, or b) placing a paragraph inside a batch of data drawn from the training corpus during inference and calculating the mean and the standard deviation on this batch. We also experimented with the instance normalization [Ulyanov *et al.*, 2016], which performs the normalization of features of a single instances, rather than of a whole minibatch. We have found that the instance normalization improved greatly upon the baseline model with no normalization, but performed worse than batch normalization.

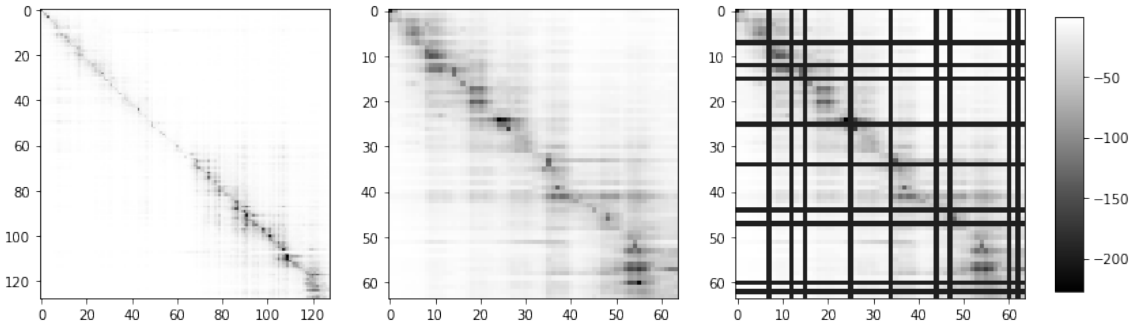
BRCA has been introduced with linear layers in the post-fix/prefix groups of the encoder/decoder. In our experiments, removing those layers from the vanilla BRCA lowered accuracy by a few percentage points. Conversely, our model benefits from not having linear layers. We observed faster convergence and better accuracy without them, while reducing the number of parameters from 23.4 million to 6.67 million.

5.6 Auto-Encoding Performance

Our training setup is comparable with that of BRCA [Zhang and LeCun, 2018]. In each epoch, we randomly select 1 million sentences from the training corpus. We trained using SGD with momentum 0.5 in batches of 32 paragraphs of random length, balanced padded to $2^r = 1024$ tokens, including



(a) Input sentence: *One of Lem's major recurring themes, beginning from his very first novel, "The Man from Mars" (. . .)*



(b) Input sentence: *Typical fuel is denatured alcohol, methanol, or isopropanol (. . .)*

Figure 3: Input-output byte relations (X axis vs. Y axis) as indicated by the method of Integrated Gradients [Sundararajan *et al.*, 2017] with 50 integration points. The plots correspond to (a) 659-byte, and (b) 128-byte Wikipedia paragraphs. The leftmost plots show relations between all input-output bytes, the middle plots for the first 64 bytes. The rightmost plots also plot spaces. Dark shades indicate strong relations -.those lay along diagonals and do not cross word and phrases boundaries.

a special end-of-sequence token. The training was run for 16 epochs, and learning rate was multiplied by 0.1 every epoch after the 10th epoch. The model suffered from the exploding gradient problem [Hochreiter *et al.*, 2001], and gradient clipping stabilized the training, enabling even higher learning rates. With clipping, we were able to set the learning rate as high as 30.0, cutting down training time to as low as 5 epochs.

Figure 4 shows auto-encoding error rate on the test set by sentence length. Our best model achieved 1.5% test error, computed as average byte-level error on the English Wikipedia dataset.

Finally, we were able to train a static version of our model (i.e., with no shared weights in the recursion group) in comparable time, closing a huge gap in convergence of recursive and static models in vanilla BRCA.

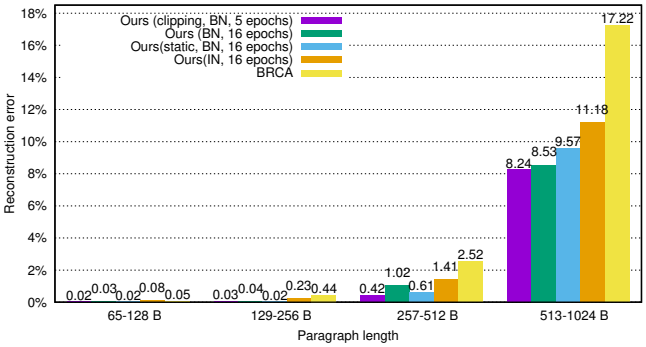


Figure 4: Decoding errors on unseen data for our best models ($N = 8$, no linear layers) with balanced input padding to a sequence of size 1024 compared with Byte-Level Recursive Convolutional Auto-Encoder (BRCA)

5.7 Generalization

We investigated which inputs influence correct predictions of the network using the method of Integrated Gradients [Sundararajan *et al.*, 2017]. We have produced two heatmaps of input-output relationships for short (128 bytes) and long (1024 bytes) paragraphs in our best model (Figure 3). In theory, a model performing identity should have a diagonal heatmap. Our model finds relations within bytes of individual words, rarely crossing word and phrases boundaries. In this sense, it fails to exploit the ordering of words. However, the order is mostly preserved in the latent vector.

Early in the training the model learns to output only spaces, which are the most common bytes in an average Wikipedia paragraph. Later during training, it learns to correctly rewrite spaces, while filling in the words with vowels, which are the most frequent non-space characters. Interestingly, the compressing behavior seems to be language-specific and triggered only by longer sequences. Figure 5 presents input sentences

```
In :  9H3cxn4RIRUnOuPymw28dxUoA060LQ3heq1diKcbiUoinkzDjxucnE3Hk7FEFwHjzcTlOrhPUp3kgt9y8VAaw1sYpjPO9N5Cv4IAn
Out:  9H3cxn4RIRUnOuPymw28dxUoA060LQ3heqddiKcbiUoinkzDjxucnE3Hk7FEFwHjzcTlOrhPUp3kgt9y8VAaw1sYpjPO9N5Cv4IAn
```

(a) Random string of characters (under 128 bytes)

```
In :  Lorsque ce m\xc3\xa9lange de cultures mondiales doit donner une signification au fa
      it d'\xc3\xaatre humain, ils r\xc3\xa9pondent avec 10,000 voix diff\xc3\xa9rentes.
Out:  Lorsque ce m\xc3\xa9lange de cultures mondiales doit donner une signification au fa
      it d'\xc3\xaatre humain, ils r\xc3\xa9pondent avec 10,000 voix diff\xc3\xa9rentes.


In :  When we first sequenced this genome in 1995, the standard of accuracy was one error per 10,000 base pairs.
Out:  When we first sequenced this genome in 1995, the standard of accuracy was one error per 10,000 base pairs.
```

(b) French and English sentences (under 256 and 128 bytes respectively)

```
In :  Lorsque ce m\xc3\xa9lange de cultures mondiales doit donner une signification au fa
      it d'\xc3\xaatre humain, ils r\xc3\xa9pondent avec 10,000 voix diff\xc3\xa9rentes.L
Out:  Larkere de Bu\xa9   lande by mortures mondiales leid dunner one mignification or Za
      an ''\xc3u   ari Rumains and pe   e   rachant open (0,000 seid disar   s   senge. M

In :  When we first sequenced this genome in 1995, the standard of accuracy was one error
      per 10,000 base pairs.When we first sequence d this genome in 1995, the standard of
Out:  When we first sequenced this genome in 1995, the standard of accuracy was one error
      per 10,000 base pair..PWew we first sequence d this genome in 1995, the standard of
```

(c) French and English sentences concatenated 4 times to form a longer input (only a prefix is shown above)

Figure 5: Auto-encoding capabilities of the model with errors marked in **bold red**. The model was trained only on English Wikipedia paragraphs. On short sequences, our model performs close to an identity function. On longer ones, it seems to correctly auto-encode only English paragraphs. Note that the model tries to map French words into English ones (`avec` $\rightarrow$ `open`, `une` $\rightarrow$ `one`). We observed a similar behavior on other languages as well.

and auto-encoded outputs of our best model, trained on English Wikipedia, for English, French and random input sequences.

6 Word-Level Sentence Encoder

Following the methods and work of [Conneau *et al.*, 2017], we apply our architecture to a practical task. Namely, we train models consisting of the recursive convolutional word-level encoder and a simple three-layer fully-connected classifier on Stanford Natural Language Inference (SNLI) corpus [Bowman *et al.*, 2015]. This dataset contains 570k sentence pairs, each one described by one of three relation labels: entailment, contradiction, and neutral. Then we test encoders on various transfer tasks measuring semantic similarities between sentences.

The encoder of each model has a similar architecture to the previously described byte-level encoder. However, instead of bytes it takes words as its input sequence. Our best encoder has $N = 8$ layers in each group. The recursive group is applied K times where 2^K is length of a padded input sequence, so that the latent vector is of the size of a word vector. We use pre-trained GloVe vectors[4] and we do not fine-tune them. We compared both fixed-length balanced, and variable length input paddings. In fixed-length padding, up to first 64 words are taken from each sentence. We also compare ensemble of our best trained model and bag-of-words as a sentence representation. Let v be the output vector of the encoder, and $u = \frac{1}{m}\sum_{i=1}^{m} e(w_i)$ be the average of word vectors of the sentence, where m is the length of the sentence, w_i is its i-th word, and $e(w)$ is the GloVe embedding of the word w. Final embedding is the sum $x = v + u$.

Table 4 presents results for word-level recursive convolutional encoder (WRCE), word-level model with fixed balanced padding (Ours), and an ensemble of our model and an average embedding of the input sequence (Ours + BoW). We compare them with a baseline model (BoW - average of GloVe vectors for words in a sentence) on SNLI and other classification tasks, SICK-Relatedness [Marelli *et al.*, 2014], and STS{12-16} tasks. The *SentEval*[5] tool was used for these experiments.

For certain tasks, especially those measuring textual similarity, which are useful in retrieval-based response generation in dialogue systems, presented models perform better than bag-of-words. However, they are still not on par with LSTM-based methods [Conneau *et al.*, 2017; Kiros *et al.*, 2015] that generate more robust embeddings. LSTM models are autoregressive and thus require slow sequential computations. They are also larger, with the InferSent model [Conneau *et al.*, 2017] having over 30 times more parameters than convolutional encoders presented in this section. In addition, our architecture can share word embedding matrices with other components of a conversational system, since word embeddings are ubiquitous in different modules of NLP systems.

In order to qualitatively assess how the results for those tasks transfer to the actual dialogue system, we have compared some retrieved responses of a simple retrieval-based

[4] `https://nlp.stanford.edu/projects/glove/`

[5] `https://github.com/facebookresearch/SentEval`

Table 4: Results for word-level sentence encoders. We compare bag-of-words (BoW), i.e. averaged word embeddings, WRCE - the encoder from Zhang and LeCun's model on word-level, our word-level model with balanced padding to 64 elements (Ours), and an ensemble of our model and BoW (Ours + BoW) for various supervised (classification accuracy) and unsupervised (Pearson/Spearman correlation coefficients) tasks.

	Model			
Task (dev/test acc%)	BoW	WRCE	Ours	Ours + BoW
SNLI	67.7 / 67.5	82.0 / 81.3	**83.8** / **83.1**	83.2 / 82.6
CR	**79.7** / 78.0	78.0 / 77.3	78.6 / 77.0	79.1 / **78.2**
MR	**77.7** / **77.0**	72.9 / 72.4	73.7 / 73.1	75.3 / 74.8
MPQA	**87.4** / 87.5	85.9 / 85.6	86.0 / 85.9	**87.4** / **87.6**
SUBJ	**91.8** / **91.4**	86.1 / 85.4	87.2 / 86.9	89.0 / 88.9
SST Bin. Class.	**80.4** / **81.4**	78.1 / 77.5	77.2 / 76.7	78.1 / 78.8
SST Fine-Grained Class.	**45.1** / **44.4**	38.3 / 40.5	40.5 / 39.3	41.9 / 41.4
TREC	**74.5** / **82.2**	67.0 / 72.4	69.2 / 71.4	71.0 / 77.4
MRPC	**74.4** / 73.2	72.4 / 71.1	73.5 / 72.5	74.1 / **73.3**
SICK-E	79.8 / 78.2	82.6 / 82.8	**83.6** / 81.9	83.2 / **83.0**
Task (correlation)	BoW	WRCE	Ours	Ours + BoW
SICK-R	0.80 / 0.72	0.85 / 0.78	**0.87** / **0.80**	0.86 / **0.80**
STS12	0.53 / 0.54	0.56 / 0.57	0.60 / 0.60	**0.62** / **0.61**
STS13	0.45 / 0.47	0.55 / 0.54	0.53 / 0.54	**0.57** / **0.58**
STS14	0.53 / 0.54	0.65 / 0.63	0.68 / **0.70**	**0.69** / 0.66
STS15	0.56 / 0.59	0.68 / 0.69	0.70 / 0.70	**0.71** / **0.72**
STS16	0.52 / 0.57	0.69 / 0.70	0.70 / 0.72	**0.71** / **0.73**

agent, which matches user utterance with a single quote from Wikiquotes [Chorowski *et al.*, 2018]. We present a comparison of our word-level sentence encoder with the bag-of-word method in response retrieval task (Figure 6). Human utterances from the training data of NIPS 2017 Conversational Challenge[6] have been selected as input utterances. We match them with the closest quote from Wikiquotes, using a method similar to the one used in *Poetwannabe* chatbot [Chorowski *et al.*, 2018]. All utterances have been filtered for foul speech (for details see [Chorowski *et al.*, 2018]), tokenized using Moses tokenizer[7], and embedded as vectors. For every user input utterance, we return the closest matching quote from Wikiquotes with respect to the cosine similarity.

7 Discussion and Future Work

The problem of efficiently producing good and robust sentence embeddings seems far from being solved. In this work, which we consider a step forward in exploration of possible tools and solutions, we analyzed and proposed improvements to the BRCA model by Xiang Zhang and Yann LeCun [2018]. With extensive usage of convolutions, our model is efficient in terms of computation and memory usage. By analyzing BRCA we were able to diagnose problems with its training, such as exploding gradients, and understand the difficulty in auto-encoding of long paragraphs, especially in the initial stage of training. Furthermore, we showed how to successfully apply batch normalization with recursive layers and investigate input-output relations with Integrated Gradients method.

The recursive convolutional architecture benefits from the ease of training and low number of parameters. Due to our realization that in the current byte-level setting, input-output relations rarely cross word boundaries, we demonstrate applicability of the architecture in a word-level setting as a sentence embedder. Furthermore, a good performance on semantic similarity tasks while using little resources demonstrates it practical usefulness for dialog systems.

Acknowledgments

The authors thank Paweł Rychlikowski and Michał Zapotoczny for fruitful discussions, and Xiang Zhang for help towards a better understanding of the original model. The research was partially supported by National Science Center (Poland) grant Sonata 8 2014/15/D/ST6/04402.

[6] http://convai.io/2017/data/

[7] https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/python-tokenizer/moses.py

User:	Hello, have you studied algebra?
BoW:	I never failed in mathematics. Before I was fifteen I had mastered differential and integral calculus.
Ours+BoW:	I never failed in mathematics. Before I was fifteen I had mastered differential and integral calculus.
User:	From what I understand it's a video game.
BoW:	Life is like a piano. What you get out of it depends on how you play it.
Ours+BoW:	From now on, I control all aspects of the game.
User:	God
BoW:	Knowledge of God is obedience to God.
Ours+BoW:	God and all attributes of God are eternal.
User:	Have you been to South Asia?
BoW:	We do not want chaos in South Africa.
Ours+BoW:	There is a wide knowledge gap between us and the developed world in the West and in Asia.
User:	How do you like Hackathon?
BoW:	Any thing you can do I can do better.
Ours+BoW:	Discover the things that you do that make you proud as hell.
User:	How do you think ?
BoW:	Any thing you can do I can do better.
Ours+BoW:	How you think matters more than what you think.
User:	I just didn't catch what was context message about.
BoW:	I read it because I was asked to explain what the truth is.
Ours+BoW:	I spend so much time thinking about all the answers to the problem... that I forget what the problem actually was.
User:	I'm an idiot
BoW:	I am an Agnostic because I am not afraid to think.
Ours+BoW:	I wish I could say I was sorry.
User:	It's classics!
BoW:	I love musical theatre and my dream is to do Once On This Island.
Ours+BoW:	No work which is destined to become a classic can look like the classics which have preceded it.
User:	So, start talking.
BoW:	Oh, ok, ok... Fair enough, yeah, rage it up. Rage all you want. Good things are coming. Good things.
Ours+BoW:	Many people talk much, and then very many people talk very much more.
User:	Technically correct
BoW:	Surely only correct understanding could lead to correct action.
Ours+BoW:	Where an opinion is general, it is usually correct.
User:	Thats why I play computer games alone.
BoW:	I have no time to play games.
Ours+BoW:	The only legitimate use of a computer is to play games.
User:	Well, can you dance?
BoW:	If I can mince , you can dance.
Ours+BoW:	Ah, so you wish to dance.
User:	What about ivy league?
BoW:	Ah wonder if anybody this side of the Atlantic has ever bought a baseball bat with playing baseball in mind.
Ours+BoW:	This is so far out of my league.

Figure 6: Sample answers of retrieval-based agents which embed sentences as either BoWs, or BoWs combined with our method

References

[Bai *et al.*, 2018] S. Bai, J. Zico Kolter, and V. Koltun. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *ArXiv e-prints*, March 2018.

[Bojanowski *et al.*, 2016] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *CoRR*, abs/1607.04606, 2016.

[Bowman *et al.*, 2015] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2015.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[Chorowski *et al.*, 2015] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *Advances in neural information processing systems*, pages 577–585, 2015.

[Chorowski *et al.*, 2018] Jan Chorowski, Adrian Lancucki, Szymon Malik, Maciej Pawlikowski, Paweł Rychlikowski, and Paweł Zykowski. A Talker Ensemble: the University of Wrocław's Entry to the NIPS 2017 Conversational Intelligence Challenge. *CoRR*, abs/1805.08032, 2018.

[Conneau *et al.*, 2017] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *CoRR*, abs/1705.02364, 2017.

[He *et al.*, 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

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

[Hochreiter *et al.*, 2001] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, and Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

[Ioffe and Szegedy, 2015] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[Joulin *et al.*, 2016] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016.

[Kiros *et al.*, 2015] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors. *CoRR*, abs/1506.06726, 2015.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, pages 1097–1105, USA, 2012. Curran Associates Inc.

[Le and Mikolov, 2014] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. *CoRR*, abs/1405.4053, 2014.

[Levitin and Reingold, 1994] Lev B Levitin and Zeev Reingold. Entropy of natural languages: Theory and experiment. *Chaos, Solitons & Fractals*, 4(5):709 – 743, 1994.

[Liu *et al.*, 2017] Huiting Liu, Tao Lin, Hanfei Sun, Weijian Lin, Chih-Wei Chang, Teng Zhong, and Alexander I. Rudnicky. Rubystar: A non-task-oriented mixture model dialog system. *CoRR*, abs/1711.02781, 2017.

[Marelli *et al.*, 2014] Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. A sick cure for the evaluation of compositional distributional semantic models. 05 2014.

[Mikolov *et al.*, 2013] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.

[Pang and Lee, 2008] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2):1–135, 2008.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[Pichl *et al.*, 2018] Jan Pichl, Petr Marek, Jakub Konrád, Martin Matulík, Hoang Long Nguyen, and Jan Sedivý. Alquist: The alexa prize socialbot. *CoRR*, abs/1804.06705, 2018.

[Ram *et al.*, 2018] Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, Eric King, Kate Bland, Amanda Wartick, Yi Pan, Han Song, Sk Jayadevan, Gene Hwang, and Art Pettigrue. Conversational AI: the science behind the alexa prize. *CoRR*, abs/1801.03604, 2018.

[Redmon *et al.*, 2015] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[Ritter *et al.*, 2011] Alan Ritter, Colin Cherry, and William B. Dolan. Data-driven response generation in social media. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 583–593, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[Serban *et al.*, 2017] Iulian Vlad Serban, Chinnadhurai Sankar, Mathieu Germain, Saizheng Zhang, Zhouhan Lin, Sandeep Subramanian, Taesup Kim, Michael Pieper, Sarath Chandar, Nan Rosemary Ke, Sai Mudumba, Alexandre de Brébisson, Jose Sotelo, Dendi Suhubdy, Vincent Michalski, Alexandre Nguyen, Joelle Pineau, and Yoshua Bengio. A deep reinforcement learning chatbot. *CoRR*, abs/1709.02349, 2017.

[Sundararajan *et al.*, 2017] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365*, 2017.

[Ulyanov *et al.*, 2016] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *CoRR*, abs/1607.08022, 2016.

[van den Oord *et al.*, 2017a] Aäron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis C. Cobo, Florian Stimberg, Norman Casagrande, Dominik Grewe, Seb Noury, Sander Dieleman, Erich Elsen, Nal Kalchbrenner, Heiga Zen, Alex Graves, Helen King, Tom Walters, Dan Belov, and Demis Hassabis. Parallel wavenet: Fast high-fidelity speech synthesis. *CoRR*, abs/1711.10433, 2017.

[van den Oord *et al.*, 2017b] Aäron van den Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals,

Koray Kavukcuoglu, George van den Driessche, Edward Lockhart, Luis C. Cobo, Florian Stimberg, Norman Casagrande, Dominik Grewe, Seb Noury, Sander Dieleman, Erich Elsen, Nal Kalchbrenner, Heiga Zen, Alex Graves, Helen King, Tom Walters, Dan Belov, and Demis Hassabis. Parallel wavenet: Fast high-fidelity speech synthesis. *CoRR*, abs/1711.10433, 2017.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[Weissenborn *et al.*, 2017] Dirk Weissenborn, Georg Wiese, and Laura Seiffe. Fastqa: A simple and efficient neural architecture for question answering. *CoRR*, abs/1703.04816, 2017.

[Yang *et al.*, 2018] Yinfei Yang, Steve Yuan, Daniel Cer, Sheng-yi Kong, Noah Constant, Petr Pilar, Heming Ge, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. Learning semantic textual similarity from conversations. *CoRR*, abs/1804.07754, 2018.

[Yusupov and Kuratov, 2017] Idris Yusupov and Yurii Kuratov. Skill-based conversational agent. 12 2017.

[Zhang and LeCun, 2018] X. Zhang and Y. LeCun. Byte-Level Recursive Convolutional Auto-Encoder for Text. *ArXiv e-prints*, February 2018.