콘볼루션 인공신경망을 이용한 단어 벡터 생성

츠나렐 재이다 ○ 장병탁
서울대학교
ceyda@bi.snu.ac.kr, btzhang@bi.snu.ac.kr

Generating Word Representations with Convolutional Neural Networks

Ceyda Cinarel ○ Byoung-Tak Zhang
Seoul National University

Abstract

Various architectures have been used for the task of language modelling. In this paper we look at a previously introduced architecture that combines CNNs, highway layers and LSTMs in and end to end trainable model. By introducing a small modification to this model we were able to achieve the same perplexity levels with less parameters.

1. Introduction

Language modelling task has been investigated using various architectures. Sequence to sequence [4] models that combine and encoder network with a decoder network have produced state of the art results on many natural language processing tasks. Variations of RNNs with LSTM and GRU units are usually used for their ability to capture long distance dependencies inherent in sentences and documents. Highway networks introduced in [5] allow unimpeded information flow across several layers. A good language model also has to capture semantic and syntactic meanings where there usually is a trade of between using word level or character level inputs. CNNs are able to learn time invariant features, temporal convolutions and temporal max pooling have been used for sentiment analysis [6] and classification [7]. An end-to-end model that combines convolutional neural networks, highway networks and stacked LSTMs was introduced in [1] for the task of language modelling setting at the time state of the art results for the PTB dataset with less parameters. We modify the architecture used in [1] by combining the convolutional layers' output with a gating function rather than just concatenating it which allows us to further reduce the number of parameters necessary to achieve the same perplexity.

2. Related Research

We specifically look at the architecture introduced in [1] which combines three different architectures; CNNs, highway networks and LSTMs. Here CNN and highway layers can be thought of as the encoder network in a sequence to sequence architecture.

It is different from other models in that it uses character level embeddings as input but makes predictions at the word level. Word representations are generated using convolutions then passed through highway network layers before the LSTM layers. We will describe in more detail how this convolutional layer works and then propose a modification to it which helps reduce the number of parameters.

Words are transformed to a $\mathbf{C}^{\mathbf{word}} \in \mathbf{R}^{\mathbf{d}\times\mathbf{l}}$ matrix using character lookup table. $\mathbf{d}$ is the character embedding size and $\mathbf{l}$ is the word length, all words are padded to be the same length. Let there be $\mathbf{k}$ number of kernels with widths $\mathbf{w_i}$ and number of feature maps $\mathbf{f}_i$ where $\mathbf{i}$ denotes the $\mathbf{i}\,\mathbf{th}$ kernel. Kernels with different widths are applied to $\mathbf{C}^{\mathbf{word}}$ for the given number of feature maps $\mathbf{f}_i$ .Then for every feature map temporal max pooling is applied resulting in a vector $\mathbf{k}_i \in \mathbf{R}^{\mathbf{f_i}}$ for each kernel. These vectors are concatenated to get the word representation $\mathbf{x}$.

$$\mathbf{x} = k_1, \ldots, k_k \quad (1)$$

Simply concatenating feature maps this way results in a

vector x of size:

$$\sum_{i=1}^{k(\#kernels)} f_i$$

Here using kernel widths of different sizes can be thought of as using different n-grams. i.e. a kernel of width 2 would correspond to 2-gram features.

3. The modified model

We hypothesize that this representation can have a lot of redundancy. Assuming that some patterns captured by the feature maps of kernel width 4 might also be captured by feature maps of kernel width 2 and so on. Inspired by [2] that uses a word based gate to combine word and character level embeddings. We suggest to use a combination of the features k_i that is modulated by a gating function, rather than the concatenation as in equation (1). In our model all the kernels of different widths have the same number of feature maps f_k.
We use the following gating function:

$$g = \text{softmax}(\tanh(Au + b))$$

Then we sum over each kernel's feature map scaled by the gate (scalar) as follows:

$$x = \sum_{i=1}^{k} g_i k_i \qquad x \in R^{f_k}$$

Here $A \in R^{kxv}$ is a weight matrix, $u \in R^v$ is the vector gate is based on, b is bias. The resulting g is a vector of size k (number of kernels). To get vector u we apply convolutions with kernel width 1 on C^{word} for 200 feature maps and use the vector after the temporal max pooling layer as $u \in R^v$. Therefore here v equals 200. Number of feature maps $v = 200$ was chosen on validation set. At first we had used word embeddings, with a lookup table, as the vector u but even though it performed well enough it didn't reduce the number of parameters as much as the current method.

2. Experiments

We used the preprocessed version of the English Penn Treebank dataset [3], the same dataset used in [1] for comparability. The vocabulary size is fixed at 10000 and the remaining words are replaced with the unknown token. Preprocessing also includes replacing numbers with a single token, removing punctuation and conversion to lower-case letters. The whole document is treated as one sequence, therefore methods such as shuffling batches shouldn't be used.

The code used for the experiments[1] was modified from the original code[2] used in [1]
The models are compared based on the test set perplexity, lower is better. For a sequence of words $[w_1, \ldots, w_T]$ perplexity is calculated as follows:

$$\text{Perplexity (PPL)} = \exp(\frac{NLL}{T})$$

Where negative log-likelihood (NLL) is the objective function:

$$NLL = -\sum_{t=1}^{T} \log P(w_t | w_{1:t-1})$$

2.1 Using the same LSTM and highway layers

[1] Reports results on two models of different sizes, small and large (denoted with *). Since our modification was only to the CNN layer we kept the hyperparameters related to the rest of the network same, such as the LSTM size and number of highway network layers. Changing only the feature maps used as shown in Table 2. As seen in Table 1 we were able to achieve the same perplexity with 5M (~%20) less parameters for the large model and for the small model better perplexity was achieved using the same number of parameters.

	LSTM size	Highway layers	#Parameters (M)	PPL
Small *	300	1	5	92.3
Small	300	1	5	90
Large *	650	2	19	78.9
Large	650	2	14	79.6

Table 1: Results where only CNN related parameters kernels, feature maps are changed. * denotes the results from [1]

	#Feature Maps (f_k)	Kernel widths
Small*	[25,50,75,100,125,150] =525	[1,2,3,4,5,6]
Large*	[50,100,150,200, 200,200,200]=1100	[1,2,3,4,5,6,7]
Small	(Gate 200), 500	[2,3,4]
Large	(Gate 200), 600	[2,3,4,5]

Table 2: The hyperparameters used in CNN layer for experiments in Table 1

1 https://github.com/snu-ceyda/lstm-char-cnn
2 https://github.com/yoonkim/lstm-char-cnn

2.2 Further hyperparameter search

We also tried models with different LSTM sizes, number of highway layers, character vector size, batch size, kernel widths and number of feature maps. Using a smaller LSTM layer we were able to achieve the same perplexity as seen in Table 3.

	PPL	#Para meters	LSTM size	Highway layers	Kernel widths
MinPPL Large	79.0	14M	600	2	[1,3,5,7,9] f_k = 700 $d = 20$
MinPPL Small	88	5M	300	2	[2,3,4] f_k = 500

Table 3: Minimum perplexity achieved with the lowest number of parameters

As the number of different width kernels increased the feature map size was also increased, which is intuitive since otherwise a fixed size vector would have to encode more features.

The number of highway layers was crucial, as also reported in [1]. When no highway layers were used the models always performed worse. No gain was observed for using more than 2 highway layers. Increasing the LSTM size more than ~700 resulted in inferior results, due to overfitting.

We also did some preliminary experiments using WikiText-2 dataset [8] which is over two times larger than the PTB dataset using punctuation, numbers and upper & lower case letters. Word vocabulary size is 33276 and character vocabulary size is 284.

LSTM size is 750, using 2 highway layers, character vector size is 50, sequence length is 40 and batch size is 40 for the gated model.

	PPL	#Para meters	Kernel widths & Feature Maps
Gated Model	96.1	36M	[1,2,3,5,7,9] f_k = 700 v =200
[10]Pointer Sentinel Mixture	96.3	20M	N/A

Table 4: Results on WikiText-2 dataset

A more through search of the hyperparameter space might yield even better results.

3. Future Research Direction

Our current gating function scales each feature map in vector k_i for a given kernel width the same amount. i.e. all 2-gram features are scaled the same amount. Instead a fine grained gating similar to [9] can be applied to get a better mix of features.

Different choices of vector u in the gating function is also worth exploring, so far we have used word embeddings and kernel width=1 convolved feature maps.

References

[1] Kim, Yoon, et al. "Character-aware neural language models." Thirtieth AAAI Conference on Artificial Intelligence (2016)

[2] Miyamoto, Yasumasa, and Kyunghyun Cho. "Gated word-character recurrent language model." arXiv preprint arXiv:1606.01700 (2016)

[3] Mikolov, Tomas, et al. "Recurrent neural network based language model." Interspeech. Vol. 2. (2010)

[4] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems. (2014)

[5] Srivastava, Rupesh Kumar, Klaus Greff, and Jürgen Schmidhuber. "Highway networks." arXiv preprint arXiv:1505.00387 (2015)

[6] Kim, Yoon. "Convolutional neural networks for sentence classification." arXiv preprint arXiv:1408.5882 (2014)

[7] Zhang, Xiang, Junbo Zhao, and Yann LeCun. "Character-level convolutional networks for text classification." Advances in neural information processing systems. (2015)

[8] Merity, Stephen, et al. "Pointer Sentinel Mixture Models." arXiv preprint arXiv:1609.07843 (2016)

[9] Yang, Zhilin, et al. "Words or Characters? Fine-grained Gating for Reading Comprehension." arXiv preprint arXiv:1611.01724 (2016)

[10] Merity, Stephen, et al. "Pointer Sentinel Mixture Models." arXiv preprint arXiv:1609.07843 (2016)