Larger-Context Language Modelling with Recurrent Neural Network[*]

Tian Wang
Center for Data Science
New York University
t.wang@nyu.edu

Kyunghyun Cho
Courant Institute of Mathematical Sciences
and Center for Data Science
New York University
kyunghyun.cho@nyu.edu

Abstract

In this work, we propose a novel method to incorporate corpus-level discourse information into language modelling. We call this larger-context language model. We introduce a late fusion approach to a recurrent language model based on long short-term memory units (LSTM), which helps the LSTM unit keep intra-sentence dependencies and inter-sentence dependencies separate from each other. Through the evaluation on four corpora (IMDB, BBC, Penn TreeBank, and Fil9), we demonstrate that the proposed model improves perplexity significantly. In the experiments, we evaluate the proposed approach while varying the number of context sentences and observe that the proposed late fusion is superior to the usual way of incorporating additional inputs to the LSTM. By analyzing the trained larger-context language model, we discover that content words, including nouns, adjectives and verbs, benefit most from an increasing number of context sentences. This analysis suggests that larger-context language model improves the unconditional language model by capturing the theme of a document better and more easily.

1 Introduction

The goal of language modelling is to estimate the probability distribution of various linguistic units, e.g., words, sentences (Rosenfeld, 2000). Among the earliest techniques were count-based n-gram language models which intend to assign the probability distribution of a given word observed after a fixed number of previous words. Later Bengio et al. (2003) proposed feed-forward neural language model, which achieved substantial improvements in perplexity over count-based language models. Bengio et al. showed that this neural language model could simultaneously learn the conditional probability of the latest word in a sequence as well as a vector representation for each word in a predefined vocabulary.

Recently recurrent neural networks have become one of the most widely used models in language modelling (Mikolov et al., 2010). Long short-term memory unit (LSTM, Hochreiter and Schmidhuber, 1997) is one of the most common recurrent activation function. Architecturally, the memory state and output state are explicitly separated by activation gates such that the vanishing gradient and exploding gradient problems described in Bengio et al. (1994) is avoided. Motivated by such gated model, a number of variants of RNNs (e.g. Cho et al. (GRU, 2014b), Chung et al. (GF-RNN, 2015)) have been designed to easily capture long-term dependencies.

When modelling a corpus, these language models assume the mutual independence among sentences, and the task is often reduced to assigning a probability to a single sentence. In this work, we propose a method to incorporate corpus-level discourse dependency into neural language model. We call this larger-context language model. It models the influence of context by defining a conditional probability in the form of $P(w_n|w_{1:n-1}, S)$, where $w_1, ..., w_n$ are words from the same sentence, and S represents the context which consists a number of previous sentences of arbitrary length.

We evaluated our model on four different corpora (IMDB, BBC, Penn TreeBank, and Fil9). Our experiments demonstrate that the proposed larger-context language model improve perplex-

[*]Recently, (Ji et al., 2015) independently proposed a similar approach.

ity for sentences, significantly reducing per-word perplexity compared to the language models without context information. Further, through Part-Of-Speech tag analysis, we discovered that content words, including nouns, adjectives and verbs, benefit the most from increasing number of context sentences. Such discovery led us to the conclusion that larger-context language model improves the unconditional language model by capturing the theme of a document.

To achieve such improvement, we proposed a late fusion approach, which is a modification to the LSTM such that it better incorporates the discourse context from preceding sentences. In the experiments, we evaluated the proposed approach against early fusion approach with various numbers of context sentences, and demonstrated the late fusion is superior to the early fusion approach.

Our model explores another aspect of context-dependent recurrent language model. It is novel in that it also provides an insightful way to feed information into LSTM unit, which could benefit all encoder-decoder based applications.

2 Statistical Language Modelling with Recurrent Neural Network

Given a document $D = (S_1, S_2, \dots, S_L)$ which consists of L sentences, statistical language modelling aims at computing its probability $P(D)$. It is often assumed that each sentence in the whole document is mutually independent from each other:

$$P(D) \approx \prod_{l=1}^{L} P(S_l). \tag{1}$$

We call this probability (before approximation) a *corpus-level probability*. Under this assumption of mutual independence among sentences, the task of language modelling is often reduced to assigning a probability to a single sentence $P(S_l)$.

A sentence $S_l = (w_1, w_2, \dots, w_{T_l})$ is a variable-length sequence of words or tokens. By assuming that a word at any location in a sentence is largely predictable by preceding words, we can rewrite the sentence probability into

$$P(S) = \prod_{t=1}^{T_l} p(w_t|w_{<t}), \tag{2}$$

where $w_{<t}$ denotes all the preceding words. We call this a *sentence-level probability*.

This rewritten probability expression can be either directly modelled by a recurrent neural network (Mikolov et al., 2010) or further approximated as a product of n-gram conditional probabilities such that

$$P(S) \approx \prod_{t=1}^{T_l} p(w_t|w_{t-(n-1)}^{t-1}), \tag{3}$$

where $w_{t-(n-1)}^{t-1} = (w_{t-(n-1)}, \dots, w_{t-1})$. The latter is called *$n$-gram language modelling*.

A recurrent language model is composed of two functions–transition and output functions. The transition function reads one word w_t and updates its hidden state such that

$$\mathbf{h}_t = \phi\left(w_t, \mathbf{h}_{t-1}\right), \tag{4}$$

where $\mathbf{h}_0$ is an all-zero vector. ϕ is a recurrent activation function. For more details on widely-used recurrent activation units, we refer the reader to (Jozefowicz et al., 2015; Greff et al., 2015).

At each timestep, the output function computes the probability over all possible *next* words in the vocabulary V. This is done by

$$p(w_{t+1} = w'|w_1^t) \propto \exp\left(g_{w'}(\mathbf{h}_t)\right). \tag{5}$$

g is commonly an affine transformation:

$$g(\mathbf{h}_t) = \mathbf{W}_o\mathbf{h}_t + \mathbf{b}_o,$$

where $\mathbf{W}_o \in \mathbb{R}^{|V|\times d}$ and $\mathbf{b}_o \in \mathbb{R}^{|V|}$.

The whole model is trained by maximizing the log-likelihood of a training corpus often using stochastic gradient descent with backpropagation through time (see, e.g., Rumelhart et al., 1988).

This conventional approach to statistical language modelling often treats every sentence in a document to be independent from each other This is often due to the fact that downstream tasks, such as speech recognition and machine translation, are done sentence-wise. In this paper, we ask how strong an assumption this is, how much impact this assumption has on the final language model quality and how much gain language modelling can get by making this assumption less strong.

Long Short-Term Memory Here let us briefly describe a long short-term memory unit which is widely used as a recurrent activation function ϕ (see Eq. (4)) for language modelling (see, e.g., Graves, 2013).

A layer of long short-term memory (LSTM) unit consists of three gates and a single memory cell. They are computed by

$$\begin{aligned}\mathbf{i}_t =& \sigma\left(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{b}_i\right)\\ \mathbf{o}_t =& \sigma\left(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{b}_o\right)\\ \mathbf{f}_t =& \sigma\left(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{b}_f\right),\end{aligned}$$

where σ is a sigmoid function. $\mathbf{x}_t$ is the input at time t. The memory cell is computed by

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh\left(\mathbf{W}_c\mathbf{x} + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c\right),$$

where $\odot$ is an element-wise multiplication. This adaptive leaky integration of the memory cell allows the LSTM to easily capture long-term dependencies in the input sequence.

The output, or the activation of this LSTM layer, is then computed by $\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t)$.

3 Larger-Context Language Modelling

In this paper, we aim not at improving the sentence-level probability estimation $P(S)$ (see Eq. (2)) but at improving the corpus-level probability $P(D)$ from Eq. (1) directly. One thing we noticed at the beginning of this work is that it is not necessary for us to make the assumption of mutual independence of sentences in a corpus. Rather, similarly to how we model a sentence probability, we can loosen this assumption by

$$P(D) \approx \prod_{l=1}^{L} P(S_l|S_{l-n}^{l-1}), \tag{6}$$

where $S_{l-n}^{l-1} = (S_{l-n}, S_{l-n+1}, \ldots, S_{l-1})$. n decides on how many preceding sentences each conditional sentence probability conditions on, similarly to what happens with a usual n-gram language modelling.

From the statistical modelling's perspective, estimating the corpus-level language probability in Eq. (6) is equivalent to build a statistical model that approximates

$$P(S_l|S_{l-n}^{l-1}) = \prod_{t=1}^{T_l} p(w_t|w_{<t}, S_{l-n}^{l-1}), \tag{7}$$

similarly to Eq. (2). One major difference from the existing approaches to statistical language modelling is that now each conditional probability of a next word is conditioned not only on the preceding words in the same sentence, but also on the $n-1$ preceding *sentences*.

A conventional, count-based n-gram language model is not well-suited due to the issue of data sparsity. In other words, the number of rows in the table storing n-gram statistics will explode as the number of possible sentence combinations grows exponentially with respect to both the vocabulary size, each sentence's length and the number of context sentences.

Either neural or recurrent language modelling however does not suffer from this issue of data sparsity. This makes these models ideal for modelling the *larger-context sentence probability* in Eq. (7). More specifically, we are interested in adapting the recurrent language model for this.

In doing so, we answer two questions in the following subsections. First, there is a question of how we should represent the context sentences S_{l-n}^{l-1}. We consider two possibilities in this work. Second, there is a large freedom in how we build a recurrent activation function to be conditioned on the context sentences. We also consider two alternatives in this case.

3.1 Context Representation

A sequence of preceding sentences can be represented in many different ways. Here, let us describe two alternatives we test in the experiments.

The first representation is to simply bag all the words in the preceding sentences into a single vector $\mathbf{s} \in [0,1]^{|V|}$. Any element of $\mathbf{s}$ corresponding to the word that exists in one of the preceding sentences will be assigned the frequency of that word, and otherwise 0. This vector is multiplied from left by a matrix $\mathbf{P}$ which is tuned together with all the other parameters: $\mathbf{p} = \mathbf{P}\mathbf{s}$. We call this representation $\mathbf{p}$ a *bag-of-words (BoW) context.*

Second, we try to represent the preceding context sentences as a sequence of bag-of-words. Each bag-of-word $\mathbf{s}_j$ is the bag-of-word representation of the j-th context sentence, and they are put into a sequence $(\mathbf{s}_{l-n}, \ldots, \mathbf{s}_{l-1})$. Unlike the first BoW context, this allows us to incorporate the order of the preceding context sentences.

This sequence of BoW vectors are read by a recurrent neural network which is separately from the one used for modelling a sentence (see Eq. (4).) We use LSTM units as recurrent activations, and for each context sentence in the sequence, we get $\mathbf{z}_t = \phi\left(\mathbf{x}_t, \mathbf{z}_{t-1}\right)$, for $t = l - n, \ldots, l-1$. We set the last hidden state $\mathbf{z}_{l-1}$ of this *context recurrent neural network* as the context vector $\mathbf{p}$.

Attention-based Context Representation The sequence of BoW vectors can be used in a bit different way from the above. Instead of a unidirectional recurrent neural network, we first use a bidirectional recurrent neural network to read the sequence. The forward recurrent neural network reads the sequence as usual in a forward direction, and the reverse recurrent neural network in the opposite direction. The hidden states from these two networks are then concatenated for each context sentence in order to form a sequence of annotation vectors $(\mathbf{z}_{l-n}, \ldots, \mathbf{z}_{l-1})$.

Unlike the other approaches, in this case, the context vector $\mathbf{p}$ differs for each word w_t in the current sentence, and we denote it by $\mathbf{p}_t$. The context vector $\mathbf{p}_t$ for the t-th word is computed as the weighted sum of the annotation vectors:

$$\mathbf{p}_t = \sum_{l'=l-n}^{l-1} \alpha_{t,l'} \mathbf{z}_{l'},$$

where the attention weight $\alpha_{t,l'}$ is computed by

$$\alpha_{t,l'} = \frac{\exp \text{score}\left(\mathbf{z}_{l'}, \mathbf{h}_t\right)}{\sum_{k=l-n}^{l-1} \exp \text{score}\left(\mathbf{z}_k, \mathbf{h}_t\right)}.$$

$\mathbf{h}_t$ is the hidden state of the recurrent language model of the current sentence from Eq. (5). The scoring function $\text{score}(\mathbf{z}_{l'}, \mathbf{h}_t)$ returns a relevance score of the l'-th context sentence w.r.t. $\mathbf{h}_t$.

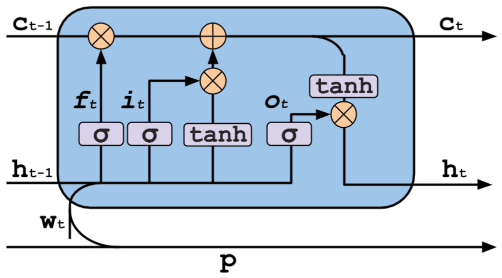
3.2 Conditional LSTM

Early Fusion Once the context vector $\mathbf{p}$ is computed from the n preceding sentences, we need to feed this into the sentence-level recurrent language model. One most straightforward way is to simply consider it as an input at every time step such that

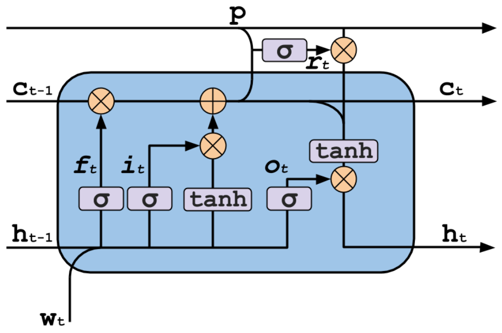
$$\mathbf{x} = \mathbf{E}^\top \mathbf{w}_t + \mathbf{W}_p \mathbf{p},$$

where $\mathbf{E}$ is the word embedding matrix that transforms the one-hot vector of the t-th word into a continuous word vector. We call this approach an *early fusion* of the context.

Late Fusion In addition to this approach, we propose here a modification to the LSTM such that it better incorporates the context from the preceding sentences (summarized by $\mathbf{p}_t$.) The basic idea is to keep dependencies within the sentence being modelled (*intra-sentence dependencies*) and those between the preceding sentences and the current sent (*inter-sentence dependencies*) separately from each other.



(a) Early Fusion



(b) Late Fusion

Figure 1: Proposed fusion methods

We let the memory cell $\mathbf{c}_t$ of the LSTM to model intra-sentence dependencies. This simply means that there is no change to the existing formulation of the LSTM.

The inter-sentence dependencies are reflected on the interaction between the memory cell $\mathbf{c}_t$, which models intra-sentence dependencies, and the context vector $\mathbf{p}$, which summarizes the n preceding sentences. We model this by first computing the amount of influence of the preceding context sentences as

$$\mathbf{r}_t = \sigma\left(\mathbf{W}_r\left(\mathbf{W}_p \mathbf{p}\right) + \mathbf{W}_r \mathbf{c}_t + \mathbf{b}_r\right).$$

This vector $\mathbf{r}_t$ controls the strength of each of the elements in the context vector $\mathbf{p}$. This amount of influence from the n preceding sentences is decided based on the currently captured intra-sentence dependency structures and the preceding sentences.

This controlled context vector $\mathbf{r}_t \odot (\mathbf{W}_p \mathbf{p})$ is used to compute the output of the LSTM layer:

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t + \mathbf{r}_t \odot \left(\mathbf{W}_p \mathbf{p}\right)\right).$$

This is illustrated in Fig. 1 (b).

We call this approach a *late fusion*, as the effect of the preceding context is fused together with the intra-sentence dependency structure in the later stage of the recurrent activation.

Late fusion is a simple, but effective way to mitigate the issue of vanishing gradient in corpus-level language modelling. By letting the context representation flow without having to pass through saturating nonlinear activation functions, it provides a linear path through which the gradient for the context flows easily.

4 Related Work

Context-dependent Language Model This possibility of extending a neural or recurrent language modeling to incorporate larger context was explored earlier. Especially, (Mikolov and Zweig, 2012) proposed an approach, called context-dependent recurrent neural network language model, very similar to the proposed approach here. The basic idea of their approach is to use a topic distribution, represented as a vector of probabilities, of previous n *words* when computing the hidden state of the recurrent neural network each time.

There are three major differences in the proposed approach from the work by Mikolov and Zweig (2012). First, the goal in this work is to explicitly model *preceding sentences* to better approximate the corpus-level probability (see Eq. (6)) rather than to get a better context of the current sentence. Second, Mikolov and Zweig (2012) use an external method, such as latent Dirichlet allocation (Blei et al., 2003) or latent semantics analysis (Dumais, 2004) to extract a feature vector, whereas we learn the whole model, including the context vector extraction, end-to-end. Third, we propose a late fusion approach which is well suited for the LSTM units which have recently been widely adopted many works involving language models (see, e.g., Sundermeyer et al., 2015). This late fusion is later shown to be superior to the early fusion approach.

Dialogue Modelling with Recurrent Neural Networks A more similar model to the proposed larger-context recurrent language model is a hierarchical recurrent encoder decoder (HRED) proposed recently by Serban et al. (2015). The HRED consists of three recurrent neural networks to model a dialogue between two people from the perspective of one of them, to which we refer as a speaker. If we consider the last utterance of the speaker, the HRED is a larger-context recurrent language model with early fusion.

Aside the fact that the ultimate goals differ (in their case, dialogue modelling and in our case, document modelling), there are two technical differences. First, they only test with the early fusion approach. We show later in the experiments that the proposed late fusion gives a better language modelling quality than the early fusion. Second, we use a sequence of bag-of-words to represent the preceding sentences, while the HRED a sequence of sequences of words. This allows the HRED to potentially better model the order of the words in each preceding sentence, but it increases computational complexity (one more recurrent neural network) and decreases statistical efficient (more parameters with the same amount of data.)

Skip-Thought Vectors Perhaps the most similar work is the skip-thought vector by Kiros et al. (2015). In their work, a recurrent neural network is trained to read a current sentence, as a sequence of words, and extract a so-called skip-thought vector of the sentence. There are two other recurrent neural networks which respectively model preceding and following sentences. If we only consider the prediction of the following sentence, then this model becomes a larger-context recurrent language model which considers a single preceding sentence as a context.

As with the other previous works we have discussed so far, the major difference is in the ultimate goal of the model. Kiros et al. (2015) fully focused on using their model to extract a good, generic sentence vector, while in this paper we are focused on obtaining a good language model. There are less major technical differences. First, the skip-thought vector model conditions only on the immediate preceding sentence, while we extend this to multiple preceding sentences. Second, similarly to the previous works by Mikolov and Zweig (2012), the skip-thought vector model only implements early fusion.

Neural Machine Translation Neural machine translation is another related approach (Forcada and Ñeco, 1997; Kalchbrenner and Blunsom, 2013; Cho et al., 2014b; Sutskever et al., 2014; Bahdanau et al., 2014). In neural machine translation, often two recurrent neural networks are used. The first recurrent neural network, called an encoder, reads a source sentence, represented as a sequence of words in a source language, to form a context vector, or a set of context vectors. The other recurrent neural network, called a decoder, then, models the target translation *conditioned on* this source context.

This is similar to the proposed larger-context recurrent language model, if we consider the source sentence as a preceding sentence in a corpus. The major difference is in the ultimate application, machine translation vs. language modelling, and technically, the differences between neural machine translation and the proposed larger-context language model are similar to those between the HRED and the larger-context language model.

Context-Dependent Question-Answering Models Context-dependent question-answering is a task in which a model is asked to answer a question based on the facts from a natural language paragraph. The question and answer are often formulated as filling in a missing word in a query sentence (Hermann et al., 2015; Hill et al., 2015). This task is closely related to the larger-context language model we proposed in this paper in the sense that its goal is to build a model to learn

$$p(q_k|q_{<k}, q_{>k}, D), \tag{8}$$

where q_k is the missing k-th word in a query Q, and $q_{<k}$ and $q_{>k}$ are the context words from the query. D is the paragraph containing facts about this query. It is explicitly constructed so that the query q does not appear in the paragraph D.

It is easy to see the similarity between Eq. (8) and one of the conditional probabilities in the r.h.s. of Eq. (7). By replacing the context sentences S_{l-n}^{l-1} in Eq. (7) with D in Eq. (8) and conditioning w_t on both the preceding and following words, we get a context-dependent question-answering model. In other words, the proposed larger-context language model can be used for context-dependent question-answering, however, with computational overhead. The overhead comes from the fact that for every possible answer the conditional probability completed query sentence must be evaluated.

5 Experimental Settings

5.1 Models

There are six possible combinations of the proposed methods. First, there are two ways of representing the context sentences; (1) bag-of-words (BoW) and (2) a sequence of bag-of-words (SeqBoW), from Sec. 3.1. There are two separate ways to incorporate the SeqBoW; (1) with attention mechanism (ATT) and (2) without it. Then, there are two ways of feeding the context vector into the main recurrent language model (RLM); (1) early fusion (EF) and (2) late fusion (LF), from Sec. 3.2. We will denote them by

1. RLM-BoW-EF-n
2. RLM-SeqBoW-EF-n
3. RLM-SeqBoW-ATT-EF-n
4. RLM-BoW-LF-n
5. RLM-SeqBoW-LF-n
6. RLM-SeqBoW-ATT-LF-n

n denotes the number of preceding sentences to have as a set of context sentences. We test four different values of n; 1, 2, 4 and 8.

As a baseline, we also train a recurrent language model without any context information. We refer to this model by RLM. Furthermore, we also report the result with the conventional, count-based n-gram language model with the modified Kneser-Ney smoothing with KenLM (Heafield et al., 2013).

Each recurrent language model uses 1000 LSTM units and is trained with Adadelta (Zeiler, 2012) to maximize the log-likelihood; $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{K}\sum_{k=1}^{K} \log p(S_k|S_{k-n}^{k-1})$. We early-stop training based on the validation log-likelihood and report the perplexity on the test set using the best model according to the validation log-likelihood.

We use only those sentences of length up to 50 words when training a recurrent language model for the computational reason. For KenLM, we used all available sentences in a training corpus.

5.2 Datasets

We evaluate the proposed larger-context language model on three different corpora. For detailed statistics, see Table 1.

IMDB Movie Reviews A set of movie reviews is an ideal dataset to evaluate many different settings of the proposed larger-context language models, because each review is highly likely of a single theme (the movie under review.) A set of words or the style of writing will be well determined based on the preceding sentences.

We use the IMDB Movie Review Corpus (IMDB) prepared by Maas et al. (2011).[1] This corpus has 75k training reviews and 25k test reviews.

[1]`http://ai.stanford.edu/~amaas/data/sentiment/`

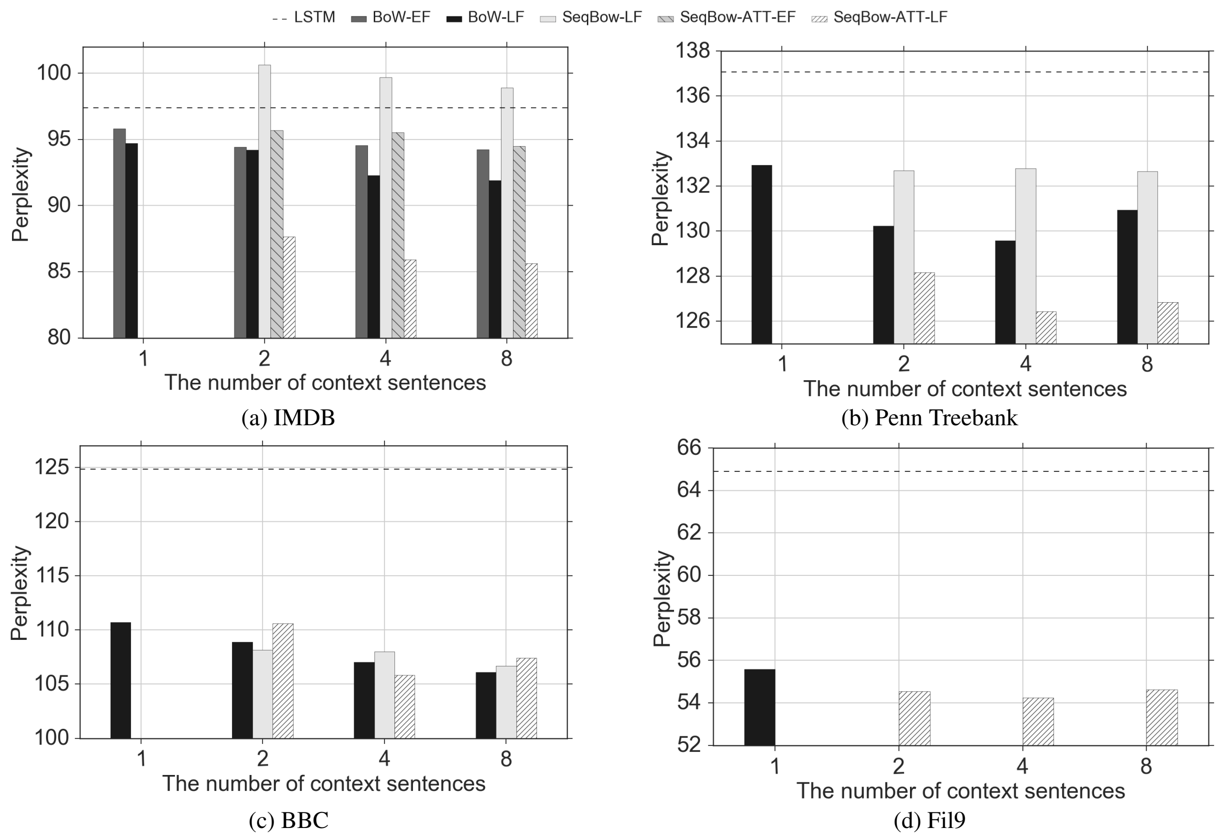


Figure 2: Corpus-level perplexity on (a) IMDB, (b) Penn Treebank, (c) BBC and (d) Fil9. The count-based 5-gram language models with Kneser-Ney smoothing respectively resulted in the perplexities of 110.20, 148, 127.32 and 65.21, and are not shown here.

We use the 30k most frequent words in the training corpus for recurrent language models.

BBC Similarly to movie reviews, each new article tends to convey a single theme. We use the BBC corpus prepared by Greene and Cunningham (2006).[2] Unlike the IMDB corpus, this corpus contains news articles which are almost always written in a formal style. By evaluating the proposed approaches on both the IMDB and BBC corpora, we can tell whether the benefits from larger context exist in both informal and formal languages. We use the 10k most frequent words in the training corpus for recurrent language models.

Both with the IMDB and BBC corpora, we did not do any preprocessing other than tokenization.[3]

Penn Treebank We evaluate a normal recurrent language model, count-based n-gram language model as well as the proposed RLM-BoW-EF-n and RLM-BoW-LF-n with varying $n = 1, 2, 4, 8$ on the Penn Treebank Corpus. We preprocess the corpus according to (Mikolov et al., 2011) and use a vocabulary of 10k words from the training corpus.

Fil9 Fil9 is a cleaned Wikipedia corpus, consisting of approximately 140M tokens, and is provided on Matthew Mahoney's website.[4] We tokenized the corpus and used the 44k most frequent words in the training corpus for recurrent language models.

6 Results and Analysis

Corpus-level Perplexity We evaluated the models, including all the proposed approaches (RLM-{BoW,SeqBoW}-{ATT,∅}-{EF,LF}-n), on the IMDB corpus. In Fig. 2 (a), we see three major trends. First, RLM-BoW, either with the early fusion or late fusion, outperforms both the count-based n-gram and recurrent language model (LSTM) regardless of the number of context sentences. Second, the improvement grows as the number n of context sentences increases, and this is most visible with the novel *late fusion*. Lastly,

[2]http://mlg.ucd.ie/datasets/bbc.html

[3]https://github.com/moses-smt/mosesdecoder/blob/master/scripts/tokenizer/tokenizer.perl

[4]http://mattmahoney.net/dc/textdata

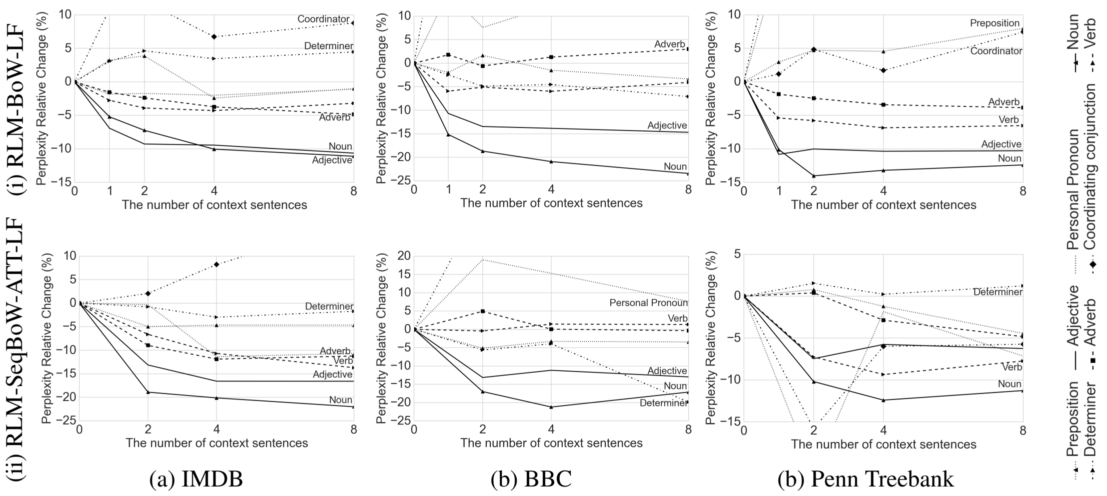


Figure 3: Perplexity per POS tag on the (a) IMDB, (b) BBC and (c) Penn Treebank corpora.

we see that the RLM-SeqBoW does not work well regardless of the fusion type (RLM-SeqBow-EF not shown), while the attention-based model (RLM-SeqBow-ATT) outperforms all the others.

After observing that the late fusion clearly outperforms the early fusion, we evaluated only RLM-{BoW,SeqBoW}-{ATT}-LF-n's on the other two corpora.

On the other two corpora, PTB and BBC, we observed a similar trend of RLM-SeqBoW-ATT-LF-n and RLM-BoW-LF-n outperforming the two conventional language models, and that this trend strengthened as the number n of the context sentences grew. We also observed again that the RLM-SeqBoW-ATT-LF outperforms RLM-SeqBoW-LF and RLM-BoW in almost all the cases.

Observing the benefit of RLM-SeqBoW-ATT-LF, we evaluated only such model on Fil9 to validate its performance on large corpus. Similar to the results on all three previous corpora, we continue to observe the advantage of RLM-SeqBoW-ATT-LF-n on Fil9 corpus.

From these experiments, the benefit of allowing larger context to a recurrent language model is clear, however, with the right choice of the context representation (see Sec. 3.1) and the right mechanism for feeding the context information to the recurrent language model (see Sec. 3.2.) In these experiments, the sequence of bag-of-words representation with attention mechanism, together with the late fusion was found to be the best choice in all four corpora.

One possible explanation on the failure of the SeqBoW representation with a context recurrent neural network is that it is simply difficult for the context recurrent neural network to compress multiple sentences into a single vector. This difficulty in training a recurrent neural network to compress a long sequence into a single vector has been observed earlier, for instance, in neural machine translation (Cho et al., 2014a). Attention mechanism, which was found to avoid this problem in machine translation (Bahdanau et al., 2014), is found to solve this problem in our task as well.

Perplexity per Part-of-Speech Tag Next, we attempted at discovering why the larger-context recurrent language model outperforms the unconditional one. In order to do so, we computed the *perplexity per part-of-speech (POS) tag*.

We used the Stanford log-linear part-of-speech tagger (Stanford POS Tagger, Toutanova et al., 2003) to tag each word of each sentence in the corpora.[5] We then computed the perplexity of each word and averaged them for each tag type separately. Among the 36 POS tags used by the Stanford POS Tagger, we looked at the perplexities of the ten most frequent tags (NN, IN, DT, JJ, RB, NNS, VBZ, VB, PRP, CC), of which we combined NN and NNS into a new tag Noun and VB and VBZ into a new tag Verb.

We show the results using the RLM-BoW-LF and RLM-SeqBoW-ATT-LF on three corpora–IMDB, BBC and Penn Treebank– in Fig. 3. We observe that the predictability, measured by the perplexity (negatively correlated), grows most for nouns (Noun) and adjectives (JJ) as the number of context sentences increases. They are followed by verbs (Verb). In other words, nouns, adjec-

[5] `http://nlp.stanford.edu/software/tagger.shtml`

	IMDB		BBC		Penn TreeBank		Fil9	
	# Sentences	# Words	# Sentences	# Words	# Sentences	# Words	# Sentences	# Words
Training	930,139	21M	37,207	890K	42,068	888K	6,619,098	115M
Validation	152,987	3M	1,998	49K	3,370	70K	825,919	14M
Test	151,987	3M	2,199	53K	3,761	79K	827,416	14M

Table 1: Statistics of IMDB, BBC, Penn TreeBank and Fil9.

tives and verbs are the ones which become more predictable by a language model given more context. We however noticed the relative degradation of quality in coordinating conjunctions (CC), determiners (DT) and personal pronouns (PRP).

It is worthwhile to note that nouns, adjectives and verbs are open-class, content, words, and conjunctions, determiners and pronouns are closed-class, function, words (see, e.g., Miller, 1999). The functions words often play grammatical roles, while the content words convey the content of a sentence or discourse, as the name indicates. From this, we may carefully conclude that the larger-context language model improves upon the conventional, unconditional language model by capturing the theme of a document, which is reflected by the improved perplexity on "content-heavy" open-class words (Chung and Pennebaker, 2007). In our experiments, this came however at the expense of slight degradation in the perplexity of function words, as the model's capacity stayed same (though, it is not necessary.)

This observation is in line with a recent finding by Hill et al. (2015). They also observed significant gain in predicting open-class, or content, words when a question-answering model, including humans, was allowed larger context.

7 Conclusion

In this paper, we proposed a method to improve language model on corpus-level by incorporating larger context. Using this model results in the improvement in perplexity on the IMDB, BBC, Penn Treebank and Fil9 corpora, validating the advantage of providing larger context to a recurrent language model.

From our experiments, we found that the sequence of bag-of-words with attention is better than bag-of-words for representing the context sentences (see Sec. 3.1), and the late fusion is better than the early fusion for feeding the context vector into the main recurrent language model (see Sec. 3.2). Our part-of-speech analysis revealed that content words, including nouns, adjectives and verbs, benefit most from an increasing number of context sentences. This analysis suggests that larger-context language model improves perplexity because it captures the theme of a document better and more easily.

To explore the potential of such a model, there are several aspects in which more research needs to be done. First, the four datasets we used in this paper are relatively small in the context of language modelling, therefore the proposed larger-context language model should be evaluated on larger corpora. Second, more analysis, beyond the one based on part-of-speech tags, should be conducted in order to better understand the advantage of such larger-context models. Lastly, it is important to evaluate the impact of the proposed larger-context models in downstream tasks such as machine translation and speech recognition.

Acknowledgments

This work is done as a part of the course DS-GA 1010-001 Independent Study in Data Science at the Center for Data Science, New York University. KC thanks Facebook, Google (Google Faculty Award 2016) and NVidia (GPU Center of Excellence 2015–2016).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research* 3:1137–1155.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *Neural Networks, IEEE Transactions on* 5(2):157–166.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research* 3:993–1022.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder-decoder approaches. In *Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation (SSST-8)*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* .

Cindy Chung and James W Pennebaker. 2007. The psychological functions of function words. *Social communication* pages 343–359.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2015. Gated feedback recurrent neural networks. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. pages 2067–2075.

Susan T Dumais. 2004. Latent semantic analysis. *Annual review of information science and technology* 38(1):188–230.

Mikel L Forcada and Ramón P Ñeco. 1997. Recursive hetero-associative memories for translation. In *Biological and Artificial Computation: From Neuroscience to Technology*, Springer, pages 453–462.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850* .

Derek Greene and Pádraig Cunningham. 2006. Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proc. 23rd International Conference on Machine learning (ICML'06)*. ACM Press, pages 377–384.

Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. 2015. Lstm: A search space odyssey. *arXiv preprint arXiv:1503.04069* .

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Sofia, Bulgaria, pages 690–696.

Karl Moritz Hermann, Tomáš Kočiskỳ, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *arXiv preprint arXiv:1506.03340* .

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2015. The goldilocks principle: Reading children's books with explicit memory representations. *arXiv preprint arXiv:1511.02301* .

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

Yangfeng Ji, Trevor Cohn, Lingpeng Kong, Chris Dyer, and Jacob Eisenstein. 2015. Document context language models. *arXiv preprint arXiv:1511.03962* .

Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. pages 2342–2350.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *EMNLP*. pages 1700–1709.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. *arXiv preprint arXiv:1506.06726* .

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pages 142–150.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*. pages 1045–1048.

Tomáš Mikolov, Stefan Kombrink, Lukáš Burget, Jan Honza Černockỳ, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, pages 5528–5531.

Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *SLT*. pages 234–239.

George A Miller. 1999. On knowing a word. *Annual Review of psychology* 50(1):1–19.

Ronald Rosenfeld. 2000. Two decades of statistical language modeling: where do we go from here. In *Proceedings of the IEEE*.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1988. Learning representations by back-propagating errors. *Cognitive modeling* 5:3.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. Hierarchical neural network generative models for movie dialogues. *arXiv preprint arXiv:1507.04808* .

Martin Sundermeyer, Hermann Ney, and Ralf Schluter. 2015. From feedforward to recurrent lstm neural networks for language modeling. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on* 23(3):517–529.

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112.

Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, pages 173–180.

Matthew D Zeiler. 2012. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701* .