A Neural Language Model for Dynamically Representing the Meanings of Unknown Words and Entities in a Discourse

Sosuke Kobayashi
Preferred Networks, Inc., Japan
`sosk@preferred.jp`

Naoaki Okazaki
Tokyo Institute of Technology, Japan
`okazaki@c.titech.ac.jp`

Kentaro Inui
Tohoku University / RIKEN, Japan
`inui@ecei.tohoku.ac.jp`

Abstract

This study addresses the problem of identifying the meaning of unknown words or entities in a discourse with respect to the word embedding approaches used in neural language models. We proposed a method for on-the-fly construction and exploitation of word embeddings in both the input and output layers of a neural model by tracking contexts. This extends the dynamic entity representation used in Kobayashi et al. (2016) and incorporates a copy mechanism proposed independently by Gu et al. (2016) and Gulcehre et al. (2016). In addition, we construct a new task and dataset called *Anonymized Language Modeling* for evaluating the ability to capture word meanings while reading. Experiments conducted using our novel dataset show that the proposed variant of RNN language model outperformed the baseline model. Furthermore, the experiments also demonstrate that dynamic updates of an output layer help a model predict reappearing entities, whereas those of an input layer are effective to predict words following reappearing entities.

1 Introduction

Language models that use probability distributions over sequences of words are found in many natural language processing applications, including speech recognition, machine translation, text summarization, and dialogue utterance generation. Recent studies have demonstrated that language models trained using neural network (Bengio et al., 2003; Mikolov et al., 2010) such as recurrent neural network (RNN) (Jozefowicz et al., 2016) and convolutional neural network (Dauphin
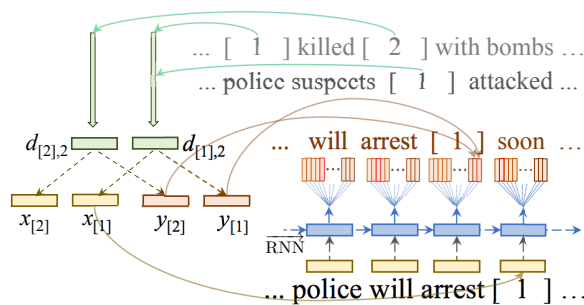

Figure 1: Dynamic Neural Text Modeling: the embeddings of unknown words, denoted by coreference indexes "[k]" are dynamically computed and used in both the input and output layers ($x_{[k]}$ and $y_{[k]}$) of a RNN language model. These are constructed from contextual information ($d_{[k],i}$) preceding the current $(i + 1)$-th sentence.

et al., 2016) achieve the best performance across a range of corpora (Mikolov et al., 2010; Chelba et al., 2014; Merity et al., 2017; Grave et al., 2017).

However, current neural language models have a major drawback: the language model works only when applied to a closed vocabulary of fixed size (usually comprising high-frequency words from the given training corpus). All occurrences of out-of-vocabulary words are replaced with a single dummy token "<unk>", showing that the word is unknown. For example, the word sequence, *Pikotaro sings PPAP on YouTube* is treated as *<unk> sings <unk> on <unk>* assuming that the words *Pikotaro*, *PPAP*, and *YouTube* are out of the vocabulary. The model therefore assumes that these words have the same meaning, which is clearly incorrect. The derivation of *meanings of unknown words* remains a persistent and nontrivial challenge when using word embeddings.

In addition, existing language models further assume that the meaning of a word is the same and universal across different documents. Neural
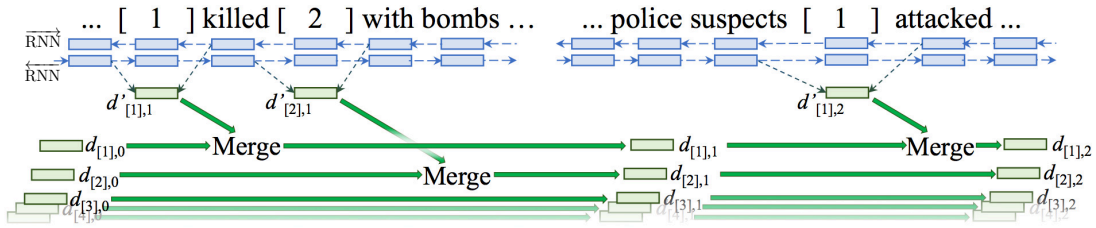
473

Figure 2: Dynamic Neural Text Modeling: the meaning representation of each unknown word, denoted by a coreference index "[k]", is inferred from the local contexts in which it occurs.

language models also make this assumption and represent all occurrences of a word with a single word vector across all documents. However, the assumption of a universal meaning is also unlikely correct. For example, the name *John* is likely to refer to different individuals in different documents. In one story, *John* may be a pianist while another *John* denoted in a second story may be an infant. A model that represents all occurrences of *John* with the same vector fails to capture the very different behavior expected from *John* as a pianist and *John* as an infant.

In this study, we address these issues and propose a novel neural language model that can build and dynamically change distributed representations of words based on the multi-sentential discourse. The idea of incorporating dynamic meaning representations into neural networks is not new. In the context of reading comprehension, Kobayashi et al. (2016) proposed a model that dynamically computes the representation of a named entity mention from the local context given by its prior occurrences in the text. In neural machine translation, the *copy mechanism* was proposed as a way of improving the handling of out-of-vocabulary words (e.g., named entities) in a source sentence (Gu et al., 2016; Gulcehre et al., 2016). We use a variant of recurrent neural language model (RNLM), that combines dynamic representation and the copy mechanism. The resulting novel model, *Dynamic Neural Text Model*, uses the dynamic word embeddings that are constructed from the context in the output and input layers of an RNLM, as shown in Figures 1 and 2.

The contributions of this paper are three-fold. First, we propose a novel neural language model, which we named the *Dynamic Neural Text Model*. Second, we introduce a new evaluation task and dataset called *Anonymized Language Modeling*. This dataset can be used to evaluate the ability of

a language model to capture word meanings from contextual information (Figure 3). This task involves a kind of one-shot learning tasks, in which the meanings of entities are inferred from their limited prior occurrences. Third, our experimental results indicate that the proposed model outperforms baseline models that use only global and static word embeddings in the input and/or output layers of an RNLM. Dynamic updates of the output layer helps the RNLM predict reappearing entities, whereas those of the input layer are effective to predict words following reappearing entities. A more detailed analysis showed that the method was able to successfully capture the meanings of words across large contexts, and to accumulate multiple context information.

2 Background

2.1 RNN Language Model

Given a sequence of N tokens of a document $D = (w_1, w_2, ..., w_N)$, an RNN language model computes the probability $p(D) = \prod_{t=1}^{N} p(w_t|w_1, ..., w_{t-1})$. The computation of each factorized probability $p(w_t|w_1, ..., w_{t-1})$ can also be viewed as the task of predicting a following word w_t from the preceding words $(w_1, ..., w_{t-1})$. Typically, RNNs recurrently compute the probability of the following word w_t by using a hidden state $\boldsymbol{h}_{t-1}$ at time step $t-1$,

$$p(w_t|w_1, ..., w_{t-1}) = \frac{\exp(\vec{\boldsymbol{h}}_{t-1}^{\mathsf{T}} \boldsymbol{y}_{w_t} + b_{w_t})}{\sum_{w \in V} \exp(\vec{\boldsymbol{h}}_{t-1}^{\mathsf{T}} \boldsymbol{y}_w + b_w)},$$
(1)

$$\vec{\boldsymbol{h}}_t = \overrightarrow{\mathrm{RNN}}(\boldsymbol{x}_{w_t}, \vec{\boldsymbol{h}}_{t-1}).$$
(2)

Here, $\boldsymbol{x}_{w_t}$ and $\boldsymbol{y}_{w_t}$ denote the input and output word embeddings of w_t respectively, V represents the set of words in the vocabulary, and b_w is a bias value applied when predicting the word w. The function $\overrightarrow{\mathrm{RNN}}$ is often replaced with

LSTM (Hochreiter and Schmidhuber, 1997) or GRU (Cho et al., 2014) to improve performance.

2.2 Dynamic Entity Representation

RNN-based models have been reported to achieve better results on the CNN QA reading comprehension dataset (Hermann et al., 2015; Kobayashi et al., 2016). In the CNN QA dataset, every named entity in each document is anonymized. This is done to allow the ability to comprehend a document using neither prior nor external knowledge to be evaluated. To capture the meanings of such anonymized entities, Kobayashi et al. (2016) proposed a new model that they named *dynamic entity representation*. This encodes the local contexts of an entity and uses the resulting context vector as the word embedding of a subsequent occurrence of that entity in the input layer of the RNN. This model: (1) constructs context vectors $d'_{e,i}$ from the local contexts of an entity e at the i-th sentence; (2) merges multiple contexts of the entity e through max pooling and produces the dynamic representation $d_{e,i}$; and (3) replaces the embedding of the entity e in the $(i+1)$-th sentence with the dynamic embedding $x_{e,i+1}$ produced from $d_{e,i}$. More formally,

$$x_{e,i+1} = W_{dc}d_{e,i} + b_e, \qquad (3)$$
$$d_{e,i} = \text{maxpooling}(d'_{e,i}, d_{e,i-1}), \qquad (4)$$
$$d'_{e,i} = \text{ContextEncoder}(e,i). \qquad (5)$$

Here, b_e denotes a bias vector, maxpooling is a function that yields the largest value from the elementwise inputs, and ContextEncoder is an encoding function. Figure 2 gives an example of the process of encoding and merging contexts from sentences. An arbitrary encoder can be used for ContextEncoder; Kobayashi et al. (2016) used bidirectional RNNs, encoding the words surrounding the entity e of a sentence in both directions. If the entity e fails to appear in the i-th sentence, the embedding is not updated, i.e., $d_{e,i} = d_{e,i-1}$.

3 Proposed Method: Dynamic Neural Text Modeling

In this section, we introduce the extension of dynamic entity representation to language modeling. From Equations 1 and 2, RNLM uses a set of word embeddings in the input layer to encode the preceding contextual words, and another set of word embeddings in the output layer to predict a word

from the encoded context. Therefore, we consider incorporating the idea of dynamic representation into the word embeddings in the output layer (y_w in Equation 1) as well as in the input layer (x_w in Equation 2; refer to Figure 1). The novel extension of dynamic representation to the output layer affects predictions made for entities that appear repeatedly, whereas that in the input layer is expected to affect the prediction of words that follow the entities.

The procedure for constructing dynamic representations of e, $d_{e,i}$ is the same as that introduced in Section 2.2. Before reading the $(i + 1)$-th sentence, the model constructs the context vectors $[d'_{e,1}, ..., d'_{e,i}]$ from the local contexts of e in every preceding sentence. Here, $d'_{e,j}$ denotes the context vector of e in the j-th sentence. ContextEncoder in the model produces a context vector d'_e for e at the t-th position in a sentence, using a bidirectional RNN[1] as follows:

$$d'_e = \text{ReLU}(W_{hd}[\vec{h}_{t-1}, \overleftarrow{h}_{t+1}] + b_d), \qquad (6)$$
$$\vec{h}_t = \overrightarrow{\text{RNN}}(x_{w_t}, \vec{h}_{t-1}), \qquad (7)$$
$$\overleftarrow{h}_t = \overleftarrow{\text{RNN}}(x_{w_t}, \overleftarrow{h}_{t+1}). \qquad (8)$$

Here, ReLU denotes the ReLU activation function (Nair and Hinton, 2010), while W_{dc} and W_{hd} correspond to learnable matrices; b_d is a bias vector. As in the RNN language model, $\vec{h}_{t-1}$ and $\overleftarrow{h}_{t+1}$ as well as their composition d'_e can capture information necessary to predict the features of the target e at the t-th word.

Following context encoding, the model merges the multiple context vectors, $[d'_{e,1}, ..., d'_{e,i}]$, into the dynamic representation $d_{e,i}$ using a merging function. A range of functions are abailable for merging multiple vectors, while Kobayashi et al. (2016) used only max pooling (Equation 4). In this study, we explored three further functions: GRU, GRU followed by ReLU ($d_{e,i} = \text{ReLU}(\text{GRU}(d'_{e,i}, d_{e,i-1}))$) and a function that selects only the latest context, i.e., $d_{e,i} = d'_{e,i}$. This comparison clarifies the effect of the accumulation of contexts as the experiments proceeded[2].

---

[1] Equations 2 and 7 are identical but do not share internal parameters.

[2] Note that merging functions are not restricted to considering two arguments (a new context and a merged past context) recurrently but can consider all vectors over the whole history $[d'_{e,1}, ..., d'_{e,i}]$ (e.g., by using attention mechanism (Bahdanau et al., 2015)). However, for simplicity, this research focuses only on the case of a function with two arguments.

Original Version
The hottest gift *this Christmas* could be *Sony's new PlayStation 2*, but good luck finding one. As *Greg Lefevre* reports, many stores have sold out of *the game* even …
Anonymized Version
the hottest gift [1] could be [2] , but good luck finding one . as [3] reports , many stores have sold out of [2] even …

Figure 3: An example document for Anonymized Language Modeling. Token "[k]" is an anonymized token that appears k-th in the entities in a document. Language models predict the next word from the preceding words, and calculate probabilities for whole word sequences.

The merging function produces the dynamic representation $d_{e,i}$ of e. In language modeling, to read the $(i + 1)$-th sentence, the model uses two dynamic word embeddings of e in the input and output layers. The input embedding x_e, used to encode contexts (Equation 2), and the output embedding y_e, used to predict the occurrence of e (Equation 1), are replaced with dynamic versions:

$$x_e = W_{dx}d_{e,i} + b_e^x, \qquad (9)$$

$$y_e = W_{dy}d_{e,i} + b_e^y, \qquad (10)$$

where W_{dx} and W_{dy} denote learnable matrices, and b_e^x and b_e^y denote learnable vectors tied to e. We can observe that a conventional RNN language model is a variant that removes the dynamic terms ($W_{dx}d_{e,i}$ and $W_{dy}d_{e,i}$) using only the static terms (b_e^x and b_e^y) to represent e. The initial dynamic representation $d_{e,0}$ is defined as a zero vector, so that the initial word embeddings (x_e and y_e) are identical to the static terms (b_e^x and b_e^y) until the point at which the first context of the target word e is observed. All parameters in the end-to-end model are learned entirely by backpropagation, maximizing the log-likelihood in the same way as a conventional RNN language model.

We can view the approach in Kobayashi et al. (2016) as a variant on the proposed method, but using the dynamic terms only in the input layer (for x_e). We can also view the copy mechanism (Gu et al., 2016; Gulcehre et al., 2016) as a variant on the proposed method, in which specific embeddings in the output layer are replaced with special dynamic vectors.

4 Anonymized Language Modeling

This study explores methods for on-the-fly capture and exploitation of the meanings of unknown words or entities in a discourse. To do this, we introduce a novel evaluation task and dataset that we called *Anonymized Language Modeling*. Figure 3 gives an example from the dataset. Briefly, the dataset anonymizes certain noun phrases, treating them as unknown words and retaining their coreference relations. This allows a language model to track the context of every noun phrase in the discourse. Other words are left unchanged, allowing the language model to preserve the context of the anonymized (unknown) words, and to infer their meanings from the known words. The process was inspired by Hermann et al. (2015), whose approach has been explored by the research on reading comprehension.

More precisely, we used the OntoNotes (Pradhan et al., 2012) corpus, which includes documents with coreferences and named entity tags manually annotated. We assigned an anonymous identifier to every coreference chain in the corpus[3] in order of first appearance[4], and replaced mentions of a coreference chain with its identifier. In our experiments, each coreference chain was given a dynamic representation. Following Mikolov et al. (2010), we limited the vocabulary to 10,000 words appearing frequently in the corpus. Finally, we inserted "<bos>" and "<eos>" tokens to mark the beginning and end of each sentence.

An important difference between this dataset and the one presented in Hermann et al. (2015) is in the way that coreferences are treated. Hermann et al. (2015) used automatic resolusion of coreferences, whereas our study made use of the manual annotations in the OntoNotes. Thus, the process of Hermann et al. (2015) introduced (intentional and unintentional) errors into the dataset. Additionally, the dataset did not assign an entity iden-

---

[3]We used documents with no more than 50 clusters, which covered more than 97% of the corpus.

[4]Following the study of Luong et al. (2015), we assigned "<unk1>", "<unk2>", ... to coreference clusters in order of first appearance.

Split	Train	Valid	Test
# of documents	2725	335	336
Avg. # of sentences	25.7	27.2	26.4
Avg. # of unique entities	15.6	16.8	15.8
Avg. # of unique entities occurring more than once	9.3	9.9	9.5
Avg. # of occurrences of an entity	3.2	3.2	3.1

Table 1: Statistics of Anonymized Language Modeling dataset.

tifier to a pronoun. In contrast, as our dataset has access to the manual annotations of coreferences, we are able to investigate the ability of the language model to capture meanings from contexts.

Dynamic updating could be applied to words in all lexical categories, including verbs, adjectives, and nouns without requiring additional extensions. However, verbs and adjectives were excluded from targets of dynamic updates in the experiments, for two reasons. First, proper nouns and nouns accounted for the majority (70%) of the low-frequency (unknown) words, followed by verbs (10%) and adjectives (9%). Second, we assumed that the meaning of a verb or adjective would shift less over the course of a discourse than that of a noun. When semantic information of unknown verbs and adjectives is required, their embeddings may be extracted from ad-hoc training on a different larger corpus. This, however, was beyond the scope of this study.

5 Experiments

5.1 Setting

An experiment was conducted to investigate the effect of *Dynamic Neural Text Model* on the *Anonymized Language Modeling* dataset. The split of dataset followed that of the original corpus (Pradhan et al., 2012). Table 1 summarizes the statistics of the dataset.

The baseline model was a typical LSTM RNN language model with 512 units. We compared three variants of the proposed model, using different applications of dynamic embedding: in the input layer only (as in Kobayashi et al. (2016)), in the output layer only, and in both the input and output layers. The context encoders were bidirectional LSTMs with 512 units, the parameters of which were not the same as those in the LSTM RNN language models. All models were trained by maximizing the likelihood of correct tokens, to

achieve best perplexity on the validation dataset[5]. Most hyper-parameters were tuned and fixed by the baseline model on the validation dataset[6].

It is difficult to adequately train the all parts of a model using only the small dataset of Anonymized Language Modeling. We therefore pretrained word embeddings and ContextEncoder (the bidirectional RNNs and matrices in Equations 6–8) on a sentence completion task in which clozes were predicted from the surrounding words in a large corpus (Melamud et al., 2016)[7]. We used the objective function with negative sampling (Mikolov et al., 2013): $\sum_e (\log \sigma(\hat{\boldsymbol{x}}_e^\intercal \boldsymbol{x}_e) + \sum_{v \in Neg}(\log \sigma(-\hat{\boldsymbol{x}}_e^\intercal \boldsymbol{x}_v)))$. Here, $\hat{\boldsymbol{x}}_e$ is a context vector predicted by ContextEncoder, $\boldsymbol{x}_e$ denotes the word embedding of a target word e appearing in the corpus, and Neg represents randomly sampled words. These pretrained parameters of ContextEncoder were fixed when the whole language model was trained on the Anonymized Language Modeling dataset. We implemented models in Python using the `Chainer` neural network library (Tokui et al., 2015). The code and the constructed dataset are publicly available[8].

5.2 Results and Analysis

5.2.1 Perplexity

Table 2 shows performance of the baseline model and the three variants of the proposed method in terms of perplexity. The table reports the mean and standard error of three perplexity values after training using three different randomly chosen initializations (we used the same convention

---

Models	(1) All	(2) Reappearing entities	(3) Following entities	(4) Non-entities
LSTM LM (Baseline) (A)	64.8±0.6	48.0±2.6	128.6±2.0	68.5±0.2
With only dynamic input (B)	62.8±0.3	42.4±1.1	109.5±1.4	**66.4±0.3**
With only dynamic output (C)	62.5±0.3	35.9±3.7	129.0±0.7	69.5±0.3
With dynamic input & output (D)	**60.7±0.2**	**34.0±1.3**	**106.8±0.6**	67.6±0.04

Table 2: Perplexities for each token group of models on the test set of Anonymized Language Modeling dataset. All values are averages with standard errors, calculated respectively by three models (trained with different random numbers). Dynamic models used GRU followed by ReLU as the merging function.

throughout this paper). Here, we discuss the proposed method using GRU followed by ReLU as the merging function, as this achieved the best perplexity (see Section 5.2.2 for a comparison of functions). We also show perplexitiy values when evaluating words of specific categories: (1) all words; (2) reappearing entity words; (3) words following entities; and (4) non-entity words.

All variants of the proposed method outperformed the baseline model. Focusing on the categories (2) and (3) highlights the roles of dynamic updates of the input and output layers. Dynamic updates of the input layer (B) had a larger improvement for predicting words following entities (3) than those of the output layer (C). In contrast, dynamic updates of the output layer (C) were quite effective for predicting reappearing entities (2) whereas those of the input layer (B) were not. These facts confirm that: dynamic updates of the input layer help a model predict words following entities by supplying on-the-fly context information; and those of the output layer are effective to predict entity words appearing multiple times.

In addition, dynamic updates of both the input and output layers (D) further improved the performance from those of either the output (C) or input (B) layer. Thus, the proposed dynamic output was shown to be compatible with dynamic input, and *vice versa*. These results demonstrated the positive effect of capturing and exploiting the context-sensitive meanings of entities.

In order to examine whether dynamic updates of the input and output embeddings capture context-sensitive meanings of entities, we present Figures 4, 5 and 6. Figure 4 depicts the perplexity of words with different positions in a document[9]. The figure confirms that the advantage of the proposed method over the baseline is more evident

[9]It is more difficult to predict tokens appearing latter in a document because the number of new (unknown) tokens increases as a model reads the document.
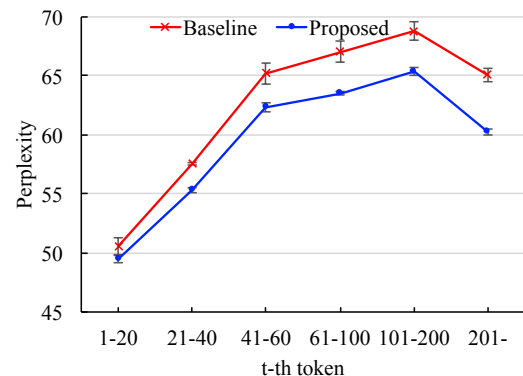

Figure 4: Perplexity of all tokens relative to the time at which they appear in the document.

especially in the latter part of documents, where repeated words are more likely to occur.

Figure 5 shows the perplexity with respect to the frequency of words t within documents. Note that the word embedding at the first occurrence of an entity is static. This figure indicates that entities appearing many times enjoy the benefit of the dynamic language model. Figure 6 visualizes the perplexity of entities with respect to the numbers of their antecedent candidates. It is clear from this figure that the proposed method is better at memorizing the semantic information of entities appearing repeatedly in documents than the baseline. These results also demonstrated the contribution of dynamic updates of word embeddings.

5.2.2 Comparison of Merging functions

Table 3 compares models with different merging functions; GRU-ReLU, GRU, max pooling, and the use of the latest context. The use of the latest context had the worst performance for all variants of the proposed method. Thus, a proper accumulation of multiple contexts is indispensable for dynamic updates of word embeddings. Although Kobayashi et al. (2016) used only max pooling as the merging function, GRU and GRU-ReLU were

Models	Merging function	# of parameters (to be finetuned)	(1) All	(2) Reappearing entities	(3) Following entities	(4) Non-entities
Only dynamic input	GRU-ReLU	18.9M (14.2M)	**62.8±0.3**	**42.4±1.1**	109.5±1.4	**66.4±0.3**
	GRU	18.9M (14.2M)	63.2±0.4	43.3±2.7	111.2±0.7	66.8±0.4
	Max pool.	17.3M (12.6M)	63.6±0.4	45.0±2.6	116.0±1.0	67.0±0.2
	Only latest	17.3M (12.6M)	64.0±0.4	44.1±1.6	127.6±0.7	67.5±0.2
Only dynamic output	GRU-ReLU	18.9M (14.2M)	62.5±0.3	**35.9±3.7**	129.0±0.7	69.5±0.3
	GRU	18.9M (14.2M)	62.6±0.2	39.0±2.0	**121.1±8.3**	69.1±0.2
	Max pool.	17.3M (12.6M)	**62.2±0.4**	41.1±1.9	126.9±1.5	**68.4±0.6**
	Only latest	17.3M (12.6M)	64.9±0.1	49.8±1.8	129.1±1.6	70.6±0.2
Dynamic input & output	GRU-ReLU	19.2M (14.4M)	**60.7±0.2**	**34.0±1.3**	**106.8±0.6**	67.6±0.04
	GRU	19.2M (14.4M)	60.9±0.3	37.5±0.3	108.9±0.8	67.2±0.4
	Max pool.	17.6M (12.9M)	**60.7±0.3**	39.5±3.4	107.5±1.3	**66.8±0.8**
	Only latest	17.6M (12.9M)	63.4±0.2	47.9±4.2	116.4±0.4	68.9±0.1
Baseline		12.3M (12.3M)	64.8±0.6	48.0±2.6	128.6±2.0	68.5±0.2

Table 3: Results for models with different merging functions on the test set of the Anonymized Language Modeling dataset, as same as in Table 2.
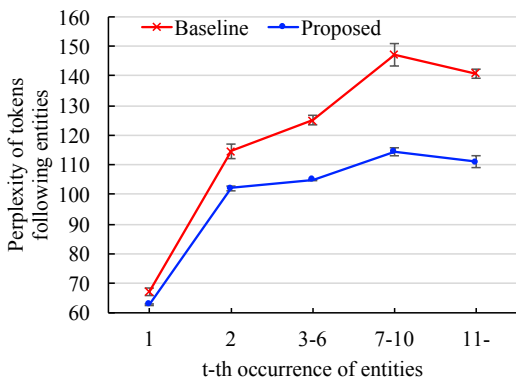

Figure 5: Perplexity of tokens following the entities relative to the time at which the entity occurs.
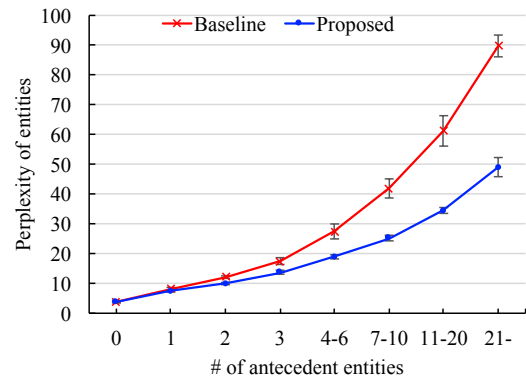

Figure 6: Perplexity of entities relative to the number of antecedent entities.

shown to be comparable in performance and superior to max pooling when predicting tokens related to entities (2) and (3).

5.2.3 Predicting Entities by Likelihood of a Sentence

In order to examine contribution of the dynamic language models on a downstream task, we conducted cloze tests for comprehension of a sentence with reappearing entities in a discourse. Given multiple preceding entities $E = \{e^+, e^1, e^2, ...\}$ followed by a cloze sentence, the models were required to predict the true antecedent e^+ which allowed the cloze to be correctly filled, among the other alternatives $E^- = \{e^1, e^2, ...\}$.

Language models solve this task by comparing the likelihoods of sentences filled with antecedent candidates in E and returning the entity with the highest likelihood of the sentence. In this experiment, the performance of a model was represented by the *Mean Quantile* (MQ) (Guu et al., 2015).

The MQ computes the mean ratio at which the model predicts a correct antecedent e^+ more likely than negative antecedents in E^-,

$$\text{MQ} = \frac{|\{e^- \in E^- : p(e^-) < p(e^+)\}|}{|E^-|}. \quad (11)$$

Here, $p(e)$ denotes the likelihood of a sentence whose cloze is filled with e. If the correct antecedent e^+ yields highest likelihood, MQ gets 1.

Table 4 reports MQs for the three variants and merging functions. Dynamic updates of the input layer greatly boosted the performance by approximately 10%, while using both dynamic input and output improved it further. In this experiment, the merging functions with GRUs outperform the others. These results demonstrated that Dynamic Neural Text Models can accumulate a new information in word embeddings and contribute to modeling the semantic changes of entities in a discourse.

479

Models	Merging func.	MQ
Baseline		.525±.001
Only dynamic input	GRU-ReLU	.630±.005
	GRU	.633±.005
	Max pool.	.617±.002
	Only latest	.600±.004
Only dynamic output	GRU-ReLU	.519±.001
	GRU	.522±.000
	Max pool.	.519±.001
	Only latest	.519±.003
Dynamic input & output	GRU-ReLU	**.642±.004**
	GRU	.637±.005
	Max pool.	.620±.002
	Only latest	.613±.002

Table 4: Mean Quantile of a true coreferent entity among antecedent entities.

6 Related Work

An approach to addressing the unknown word problem used in recent studies (Kim et al., 2016; Sennrich et al., 2016; Luong and Manning, 2016; Schuster and Nakajima, 2012) comprises the embeddings of unknown words from character embeddings or subword embeddings. Li and Jurafsky (2015) applied word disambiguation and use a sense embedding to the target word. Choi et al. (2017) captured the context-sensitive meanings of common words using word embeddings, applied through a gating function controlled by history words, in the context of machine translation. In future work, we will explore a wider range of models, to integrate our dynamic text modeling with methods that estimate the meaning of unknown words or entities from their constituents. When addressing well-known entities such as *Obama* and *Trump*, it makes sense to learn their embeddings from external resources, as well as dynamically from the preceding context in a given discourse (as in our Dynamic Neural Text Model). The integration of these two sources of information is an intriguing challenge in language modeling.

A key aspect of our model is its incorporation of the copy mechanism (Gu et al., 2016; Gulcehre et al., 2016), using dynamic word embeddings in the output layer. Independently of this study, several research groups have explored the use of variants of the copy mechanisms in language modeling (Merity et al., 2017; Grave et al., 2017; Peng and Roth, 2016). These studies, however, did not incorporate dynamic representations in the input layer. In contrast, our proposal incorporates the copy mechanism through the use

of dynamic representations in the output layer, integrating them with dynamic mechanisms in both the input and output layers by applying dynamic entity-wise representation. Our experiments have demonstrated the benefits of such integration.

Another related trend in recent studies is the use of neural network to capture the information flow of a discourse. One approach has been to link RNNs across sentences (Wang and Cho, 2016; Serban et al., 2016), while a second approach has exploited a type of memory space to store contextual information (Sukhbaatar et al., 2015; Tran et al., 2016; Merity et al., 2017). Research on reading comprehension (Kobayashi et al., 2016; Henaff et al., 2017) and coreference resolution (Wiseman et al., 2016; Clark and Manning, 2016b,a) has shown the salience of entity-wise context information. Our model could be located within such approaches, but is distinct in being the first model to make use of entity-wise context information in both the input and output layers for sentence generation.

We summarize and compare works for entity-centric neural networks that read a document. Kobayashi et al. (2016) pioneered entity-centric neural models tracking states in a discourse. They proposed *Dynamic Entity Representation*, which encodes contexts of entities and updates the states using entity-wise memories. Wiseman et al. (2016) also proposed a method for managing similar entity-wise features on neural networks and improved a coreference resolution model. Clark and Manning (2016b,a) incorporated such entity-wise representations in mention-ranking coreference models. Our paper follows Kobayashi et al. (2016) and exploits dynamic entity reprensetions in a neural language model, where dynamic reporesentations are used not only in the neural encoder but also in the decoder, applicable to various sequence generation tasks, e.g., machine translation and dialog response generation. Simultaneously with our paper, Ji et al. (2017) use dynamic entity representation in a neural language model for reranking outputs of a coreference resolution system. Yang et al. (2017) experiment language modeling with referring to internal contexts or external data. Henaff et al. (2017) focus on neural networks tracking contexts of entities, achieving the state-of-the-art result in bAbI (Weston et al., 2015), a reading comprehension task. They encode the contexts of each entity by an attention-

like gated RNN instead of using coreference links directly. Dhingra et al. (2017) also try to improve a reading comprehension model using coreference links. Similarly to our dynamic entity representation, Bahdanau et al. (2017) construct on-the-fly word embeddings of rare words from dictionary definitions.

The fisrt key component of dynamic entity representation is a function to merge more than one contexts about an entity into a consistent representation of the entity. Various choices for the function exist, e.g., max or average-pooling (Kobayashi et al., 2016; Clark and Manning, 2016b), RNN (GRU, LSTM (Wiseman et al., 2016; Yang et al., 2017) or other gated RNNs (Henaff et al., 2017; Ji et al., 2017)), or using the latest context only (without any merging) (Yang et al., 2017). This paper is the first work comparing the effects of those choices (see Section 5.2.2).

The second component is a function to encode local contexts from a given text, e.g., bidirectional RNN encoding (Kobayashi et al., 2016), unidirectional RNN used in a language model (Ji et al., 2017; Yang et al., 2017), feedforward neural network with a sentence vector and an entity's word vector (Henaff et al., 2017) or hand-crafted features with word embeddings (Wiseman et al., 2016; Clark and Manning, 2016b). This study employs bi-RNN analogously to Kobayashi et al. (2016), which can access full context with powerful learnable units.

In the task setting proposed in this study, a model must capture the meaning of a given specific word from a small number of its contexts in a given discourse. The task could also be seen as novel one-shot learning (Fei-Fei et al., 2006) of word meanings. One-shot learning for NLP like this has been little studied, with the exception of the study by Vinyals et al. (2016), which used a task in which the context of a target word is matched with a different context of the same word.

7 Conclusion

This study addressed the problem of identifying the meaning of unknown words or entities in a discourse with respect to the word embedding approaches used in neural language models. We proposed a method for on-the-fly construction and exploitation of word embeddings in both the input layer and output layer of a neural model by

tracking contexts. This extended the dynamic entity representation presented in Kobayashi et al. (2016), and incorporated a copy mechanism proposed independently by Gu et al. (2016) and Gulcehre et al. (2016). In the course of the study, we also constructed a new task and dataset, called *Anonymized Language Modeling*, for evaluating the ability of a model to capture word meanings while reading. Experiments conducted using our novel dataset demonstrated that the RNN language model variants proposed in this study outperformed the baseline model. More detailed analysis indicated that the proposed method was particularly successful in capturing the meaning of an unknown words from texts containing few instances.

References

Dzmitry Bahdanau, Tom Bosc, Stanisław Jastrzębski, Edward Grefenstette, Pascal Vincent, and Yoshua Bengio. 2017. Learning to compute word embeddings on the fly. *arXiv preprint arXiv:1706.00286*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *JOURNAL OF MACHINE LEARNING RESEARCH*, 3:1137–1155.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2014. One billion word benchmark for measuring progress in statistical language modeling. In *Proceedings of INTERSPEECH*, pages 2635–2639.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of EMNLP*.

Heeyoul Choi, Kyunghyun Cho, and Yoshua Bengio. 2017. Context-dependent word representation for neural machine translation. *Computer Speech & Language*, 45:149–160.

Kevin Clark and Christopher D. Manning. 2016a. Deep reinforcement learning for mention-ranking coreference models. In *Proceedings of EMNLP*, pages 2256–2262.

Kevin Clark and Christopher D. Manning. 2016b. Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of ACL*, pages 643–653.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. 2016. Language Modeling with Gated Convolutional Networks. *arXiv preprint arXiv:1612.08083*.

Bhuwan Dhingra, Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. 2017. Linguistic knowledge as memory for recurrent neural networks. *arXiv preprint arXiv:1703.02620*.

Li Fei-Fei, Rob Fergus, and Pietro Perona. 2006. One-shot learning of object categories. *IEEE transactions on TPAMI*, 28(4):594–611.

Edouard Grave, Armand Joulin, and Nicolas Usunier. 2017. Improving neural language models with a continuous cache. In *Proceedings of ICLR*.

Jiatao Gu, Zhengdong Lu, Hang Li, and O.K. Victor Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of ACL*, pages 1631–1640.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In *Proceedings of ACL*, pages 140–149.

Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *Proceedings of EMNLP*, pages 318–327.

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2017. Tracking the world state with recurrent entity networks. In *Proceedings of ICLR*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of NIPS*, pages 1684–1692.

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

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A. Smith. 2017. Dynamic entity representations in neural language models. In *Proceedings of EMNLP*.

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*.

Yoon Kim, Yacine Jernite, David Sontag, and Alexander M. Rush. 2016. Character-aware neural language models. In *Proceedings of AAAI*, pages 2741–2749.

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

Sosuke Kobayashi, Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2016. Dynamic entity representation with max-pooling improves machine reading. In *Proceedings of NAACL-HLT*, pages 850–855.

Jiwei Li and Dan Jurafsky. 2015. Do multi-sense embeddings improve natural language understanding? In *Proceedings of EMNLP*, pages 1722–1732.

Minh-Thang Luong and D. Christopher Manning. 2016. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of ACL*, pages 1054–1063.

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare word problem in neural machine translation. In *Proceedings of ACL*, pages 11–19.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of CoNLL*, pages 51–61.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *Proceedings of ICLR*.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proceedings of INTERSPEECH*, pages 1045–1048.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, pages 3111–3119.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of ICML*, pages 807–814. Omnipress.

Haoruo Peng and Dan Roth. 2016. Two discourse driven language models for semantics. In *Proceedings of ACL*, pages 290–300.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of CoNLL*.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *Proceedings of ICASSP*, pages 5149–5152.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of ACL*, pages 1715–1725.

Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of AAAI*, pages 3776–3783.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. 2015. End-to-end memory networks. In *Proceedings of NIPS*, pages 2440–2448.

Seiya Tokui, Kenta Oono, Shohei Hido, and Justin Clayton. 2015. Chainer: a next-generation open source framework for deep learning. In *Proceedings of Workshop on LearningSys in NIPS 28*.

Ke Tran, Arianna Bisazza, and Christof Monz. 2016. Recurrent memory networks for language modeling. In *Proceedings of NAACL-HLT*, pages 321–331.

Oriol Vinyals, Charles Blundell, Tim Lillicrap, koray kavukcuoglu, and Daan Wierstra. 2016. Matching networks for one shot learning. In *Proceedings of NIPS*, pages 3630–3638.

Tian Wang and Kyunghyun Cho. 2016. Larger-context language modelling with recurrent neural network. In *Proceedings of ACL*, pages 1319–1329.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*.

Sam Wiseman, Alexander M. Rush, and Stuart M. Shieber. 2016. Learning global features for coreference resolution. In *Proceedings of NAACL-HLT*, pages 994–1004.

Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. 2017. Reference-aware language models. In *Proceedings of EMNLP*.