Multiple-Weight Recurrent Neural Networks

Zhu Cao[1*], **Linlin Wang**[1*] and **Gerard de Melo**[2]
[1] IIIS, Tsinghua University, Beijing, China
[2] Rutgers University, New Brunswick, NJ, USA
{cao-z13, ll-wang13}@mails.tsinghua.edu.cn, gdm@demelo.org

Abstract

Recurrent neural networks (RNNs) have enjoyed great success in speech recognition, natural language processing, etc. Many variants of RNNs have been proposed, including vanilla RNNs, LSTMs, and GRUs. However, current architectures are not particularly adept at dealing with tasks involving multi-faceted contents, i.e., data with a bimodal or multimodal distribution. In this work, we solve this problem by proposing Multiple-Weight RNNs and LSTMs, which rely on multiple weight matrices to better mimic the human ability of switching between contexts. We present a framework for adapting RNN-based models and analyze the properties of this approach. Our detailed experimental results show that our model outperforms previous work across a range of different tasks and datasets.

1 Introduction

Deep learning has led to remarkable advances across a diverse range of machine learning tasks in recent years [Bengio and others, 2009]. Various kinds of deep network architectures have been proposed, including Multi-Layer Neural Networks, Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNNs). RNN-style networks have emerged as the dominant approach for sequential data such as natural language, achieving great success on tasks such as speech recognition, language modeling, and text generation.

Over the years, a number of variants of RNNs have been proposed. The well-known Long Short Term Memory (LSTM) networks [Hochreiter and Schmidhuber, 1997] were designed with the goal of mitigating the vanishing and exploding gradient problems [Bengio *et al.*, 1994; Hochreiter and Schmidhuber, 1997]. LSTM cells are able to capture longer-range associations by aggregating increasing amounts of information. They have enjoyed remarkable success on highly non-trivial tasks such as machine translation and relation extraction.

LSTMs rely on information control mechanisms in the form of gates that modulate how the state is updated, given values computed from the current input and the previous hidden state. Different gating structures and information flow mechanisms can be considered. Cho et al. [2014] presented Gated Recurrent Units (GRUs), which are able to adaptively collect dependencies at different time scales. GRUs, too, have been successfully applied across a range of tasks, including machine translation. A number of minor variations of LSTMs [Greff *et al.*, 2016] and GRUs [Jozefowicz *et al.*, 2015] have been studied. The Minimal Gated Unit (MGU) architecture [Zhou *et al.*, 2016] relies on a simplified model with just a single gate. In contrast, the recent Multi-Function Recurrent Units (Mu-FuRUs) adopt an elaborate gating mechanism that allows for additional differentiable functions as composition operations [Weissenborn and Rocktäschel, 2016]. Overall, we see that it remains important to explore the space of design choices for information flow and gating mechanisms in recurrent neural networks.

In this paper, we study such gating mechanisms focusing in particular on the important challenge of how to develop models that perform robustly on challenging multi-faceted tasks. Humans are naturally adept at adapting to different kinds of tasks. For instance, we can rapidly switch between different levels of formality in conversation, interacting with our colleague using colloquial slang in one moment, while speaking formally to a superior in the next. To this end, we develop a more adaptive RNN-style architecture that consists of multiple weight matrices so as to enhance the model's ability of handling heterogeneous types of content across different test instances.

The main contributions of this work are as follows:

1. We propose the novel paradigm of multi-weight matrix RNNs, seeking to mimic the human ability of separating and switching between different subtasks or problem facets and addressing each of them appropriately.
2. This paradigm is formalized as a framework that is readily applicable to numerous RNN-family network architectures. We analyze this framework both at an abstract level and empirically.
3. Our extensive experimental evaluation on three tasks, namely language modeling, cloze tests, and character-level word embeddings, shows that our model is successfully able to exploit the multi-faceted nature of the data, surpassing previous models on these tasks and obtaining new state-of-the-art results on some.

[*] These authors contributed equally to this work.

2 Related Work

Recurrent neural networks, a form of neural networks with self-recurrent connections, were extensively studied during the 1990s [Fausett, 1994]. It was soon observed that ordinary recurrent neural networks, in training, suffer from the problems of exploding and vanishing gradients [Bengio *et al.*, 1994]. Hence, more elaborate RNN networks were designed so as to overcome these drawbacks. LSTM networks sought to achieve this by relying on three gates that control the information flow [Hochreiter and Schmidhuber, 1997]. The more recent Gated Recurrent Unit (GRU) networks rely on two gates to capture the dependencies of different time scales adaptively [Cho *et al.*, 2014]. Apart from these well-established units, a number of extensions and variants have been put forth, including Structurally Constrained Recurrent Networks (SCRNs) [Mikolov *et al.*, 2014], IRNNs [Le *et al.*, 2015], and Minimal Gated Units (MGU) [Zhou *et al.*, 2016]. Multi-Function Recurrent Units (Mu-FuRUs) adapt the gating mechanism by allowing it to select among a set of suitable composition functions [Weissenborn and Rocktäschel, 2016]. Our work presents a framework showing how many of these RNN-style algorithms can additionally operate using multiple weight matrices.

As for the tasks we consider, important works for language modeling include the recurrent neural network language model (RNNLM) [Mikolov *et al.*, 2010], which addressed the problem that previous feedforward networks depended on fixed length contexts, which need to be specified ad hoc before training. LSTM-based language models have also achieved strong results on language modeling due to their superior capability of capturing longer-term dependencies [Hochreiter and Schmidhuber, 1997]. For dialogue systems, contextual information and dialogue interactions between speakers are important signals. Thus, several contextual language models have been proposed, such as a context-dependent RNN language model [Mikolov and Zweig, 2012], larger-context language models [Wang and Cho, 2015], and two state-of-the-art dialogue context language models [Liu and Lane, 2017]. A significant challenge for word embeddings is how to address the problem of rare and out-of-vocabulary words. Here, character-level information provides significant linguistic features for learning a good representation not only for rare words but also for morphologically complex words.

Finally, we contrast our work with other lines of works sharing seemingly similar goals. In multi-task learning [Caruana, 1997], it is known from which task each input comes, whereas in our setting, there is a single task with multi-faceted data, so the system autonomously needs to learn to choose different weight matrices. Another related task is domain adaptation [Glorot *et al.*, 2011]. Here, training occurs on one domain and then the trained model is applied (only) to another domain. Our model, in contrast, is simultaneously applied to heterogeneous multi-faceted data, without knowledge of which domains a given input instance originates from. Our architecture needs to learn on its own how to identify and handle relevant domains or facets, which can be mixed even within a single instance. Our idea of relying on multiple weight matrices is also distinct from the notion of ensemble methods, which also duplicate parts the model, and are often used for increased accuracy and robustness of neural networks. We describe the differences in Section 3.2. Finally, the differences to attention models are analyzed in Section 4.4.

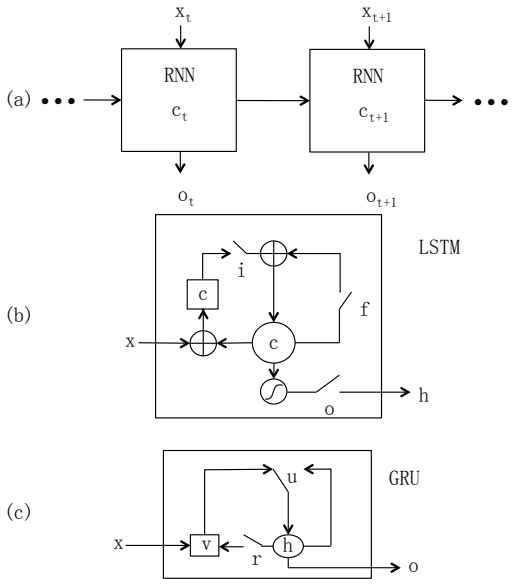

Figure 1: The structures of naïve RNN, LSTM, and GRU.

3 Multi-Weight RNN Model

In this section, we describe the details of our proposed model. We will first review model structures from past work, and then introduce the key innovations of our framework, which allows us to modify these previous architectures.

3.1 Basic RNN-based Models

An RNN-type structure consists of RNN cells of the form illustrated in Fig. 1(a). Such cells can be described in terms of their update equations as follows:

$$\begin{aligned} c_t &= f(W_{\mathrm{h}} x_t + U_{\mathrm{h}} c_{t-1} + b), \\ o_t &= g(W_{\mathrm{y}} c_t + b_{\mathrm{y}}). \end{aligned} \tag{1}$$

Here the subscript t stands for the t-th timestamp, x_t, c_t, o_t are the input vector, the state vector, and the output vector, respectively, and f, g are activation functions. For a given sequence of inputs, the repetitive application of this procedure can be described in terms of connected chains of RNNs, as illustrated in Fig. 1(a).

A naïve RNN cell computes a new state vector at every step using matrix multiplications and a non-linearity such as tanh. To overcome the naïve RNN's inability to store long-range information, several enhanced RNN cells have been proposed, among which the most well-known ones are Long Short Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997] and Gated Recurrent Unit (GRU) [Cho *et al.*, 2014] cells. An LSTM cell is illustrated in Fig. 1(b) and can be described by

the following equations:

$$\begin{aligned} i_t &= \sigma(W_{\mathrm{ix}}x_t + W_{\mathrm{ih}}h_{t-1} + b_{\mathrm{i}}), \\ f_t &= \sigma(W_{\mathrm{fx}}x_t + W_{\mathrm{fh}}h_{t-1} + b_{\mathrm{f}}), \\ o_t &= \sigma(W_{\mathrm{ox}}x_t + W_{\mathrm{oh}}h_{t-1} + b_{\mathrm{o}}), \\ c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_{\mathrm{cx}}x_t + W_{\mathrm{ch}}h_{t-1} + b), \\ h_t &= o_t \odot c_t. \end{aligned} \tag{2}$$

Here, as in a basic RNN cell, t stands for the t-th timestamp, x_t, c_t, h_t are the input vector, state vector, and hidden vector, respectively, and i_t, f_t, o_t refer to the input gate, forget gate, and output gate, respectively. Finally, the W are weight matrices, b are bias terms, and σ is the sigmoid function.

Compared to LSTMs, GRUs do not need any intermediate vector c_t. The structure of a GRU is illustrated in Fig. 1(c) and can be expressed as follows:

$$\begin{aligned} r_t &= \sigma(W_{\mathrm{rx}}x_t + W_{\mathrm{rh}}h_{t-1} + b_{\mathrm{r}}), \\ u_t &= \sigma(W_{\mathrm{ux}}x_t + W_{\mathrm{uh}}h_{t-1} + b_{\mathrm{u}}), \\ v_t &= \tanh\left[W_{\mathrm{vx}}x_t + W_{\mathrm{vr}}(r_t \odot h_{t-1}) + b_{\mathrm{v}}\right], \\ o_t &= h_t = u_t \odot h_{t-1} + (1-u_t) \odot v_t. \end{aligned} \tag{3}$$

Here, x_t, h_t, v_t, o_t are the input vector, the original activation vector, the new activation vector, and the output vector, respectively, while u_t, r_t refer to the update gate and reset gate, respectively. It has been found that GRUs and LSTMs achieve somewhat similar accuracies, but that GRUs are faster in terms of the convergence time and required training epochs [Chung *et al.*, 2014].

3.2 Our Model

Having introduced previous RNN-type models, we now present our novel multi-weight architecture. The central idea in our proposal is to allow for multiple matrices that may update the state vector. During training, the model will then learn to adaptively decide on which of these to rely on, based on the current input and the previous state at every time step. Technically, we replace each occurrence of $\tanh(Wx+b)$ by a weighted sum

$$\sum_i p_i \tanh(W_i x + b_i), \tag{4}$$

where $\tanh(\cdot)$ is the hypertangent function. Note that we are not combining the matrices W_i as $\tanh(\sum W_i)$, since that would be equivalent to $\tanh(W)$ for a single matrix W. When our modification is applied to an LSTM cell, we obtain a Multi-Weight LSTM cell with the update equations

$$\begin{aligned} i_t &= \sigma(W_{\mathrm{ix}}x_t + W_{\mathrm{ih}}h_{t-1} + b_{\mathrm{i}}), \\ f_t &= \sigma(W_{\mathrm{fx}}x_t + W_{\mathrm{fh}}h_{t-1} + b_{\mathrm{f}}), \\ o_t &= \sigma(W_{\mathrm{ox}}x_t + W_{\mathrm{oh}}h_{t-1} + b_{\mathrm{o}}), \\ c_t &= f_t \odot c_{t-1} + \sum_{i=1}^{K} p_i \odot \tanh\left(W^i_{\mathrm{cx}}x_t + W^i_{\mathrm{ch}}h_{t-1} + b^i\right), \\ h_t &= o_t \odot c_t, \end{aligned} \tag{5}$$

where K is the number of weight matrices. Our strategy is to extend the LSTM cell with a gate control mechanism to allow it to choose between two or more matrices when updating the state vector for multi-faceted data (see Fig. 2).

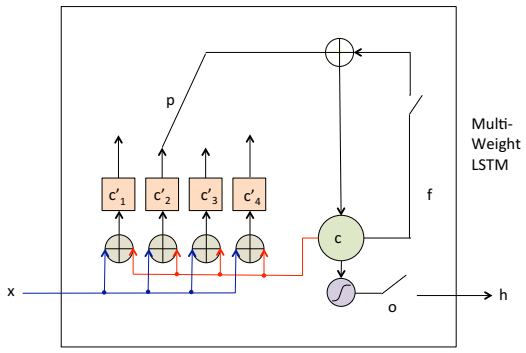

Figure 2: The structure of a multi-weight LSTM cell.

As another example, when applied to the GRU cell architecture, we obtain cell architecture of the form

$$\begin{aligned} r_t &= \sigma(W_{\mathrm{rx}}x_t + W_{\mathrm{rh}}h_{t-1} + b_{\mathrm{r}}), \\ u_t &= \sigma(W_{\mathrm{ux}}x_t + W_{\mathrm{uh}}h_{t-1} + b_{\mathrm{u}}), \\ v^i_t &= \tanh\left[W^i_{\mathrm{vx}}x_t + W^i_{\mathrm{vr}}(r_t \odot h_{t-1}) + b^i_{\mathrm{v}}\right], 1 \le i \le K \\ o_t &= h_t = u_t \odot h_{t-1} + (1-u_t) \odot \sum_{i=1}^{K} p_i v^i_t, \end{aligned} \tag{6}$$

where the v^t_i are the activation vectors for the GRU. The subscript of W indicates the weight of the connection between two gates, and the superscript of W indicates which matrix is being used. For simplicity, in later experiments, we will consider the modified LSTM cell architecture and take $K = 2$. Other cell structures improve to a similar degree.

While our architecture bears some superficial resemblance to ensemble methods, there are a few crucial differences. First, while ensemble methods aggregate the final outputs that correspond to different internal states [Zhang *et al.*, 2016], our model uses a single RNN with only one internal state. Internally, the decisions for arriving at this internal state are based on multiple matrices. This novel approach is geared at multi-faceted data. Second, while ensemble methods take an average of several model outputs, our model relies on a weighted average over the matrices, for which the weights are learnt during training.

Finally, we explain how the p_i are computed. The weights of the matrices, p_i, are also learnt using the input word and the state vector,

$$p'_i = h^i_p(x, c), \tag{7}$$

and then go through a softmax normalization,

$$p_i = \frac{\exp(p'_i)}{\sum_j \exp(p'_j)}. \tag{8}$$

A detailed description of our scheme is shown in Fig. 2. In our implementation, we choose h^i_p such that

$$p'_i = W^i_{\mathrm{px}}x + W^i_{\mathrm{pc}}c + b^i_{\mathrm{p}}. \tag{9}$$

3.3 Objective function

Since the objective function is dependent on the task, we discuss a series of three example tasks (which are explained in more detail in Section 4).

1. Language modeling: Here, the loss function is set to

$$L = -\sum_{t=1}^{T} \log p(w_t \mid w_1, \ldots, w_{t-1}). \quad (10)$$

   This loss function can be understood as maximizing the probability of a sentence.
2. Cloze tests: In this task, we set the output layer to be a conditional random field (CRF). The objective function is the conditional probability of the true answer to the cloze test, i.e.,

$$L = -\log p(w_t \mid w_1, \ldots, w_{t-1}, w_{t+1}, \ldots, w_N), \quad (11)$$

   where w_t is the blank word to be filled.
3. Character to word embedding: Here, we choose the loss function to be the cosine similarity of the ground truth word embedding and the one generated by our model from character embeddings. More precisely, we have

$$L = -\langle v_w, v_c \rangle, \quad (12)$$

   where the minus sign is placed to maximize the cosine similarity, v_w is the genuine word embedding (which serves as the ground truth target), and v_c is the word embedding generated from character embeddings.

3.4 Training

We use mini-batch stochastic gradient descent to train our model. The learning rate is controlled by AdaDelta [Zeiler, 2012]. As before, we distinguish three cases:

1) Language modeling: The parameters are learnt through the following equation

$$\frac{\partial L}{\partial \theta} = \sum_k \frac{\partial L}{\partial w_k} \frac{\partial w_k}{\partial \theta}, \quad (13)$$

where θ refers to the parameters.

2) Cloze tests: The training for cloze test tasks is similar to language modeling, except that here, only one word is being predicted. Hence the parameters are updated according to

$$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial w_t} \frac{\partial w_t}{\partial \theta}, \quad (14)$$

where θ is the parameter and w_t is the word being predicted.

3) Character to word embedding: Denote the character embeddings as $c_1, c_2, \ldots, c_n$. According to the chain rule, the parameters are updated by

$$\frac{\partial L}{\partial \theta} = \frac{\partial L}{\partial v_c} \sum_k \frac{\partial v_c}{\partial c_k} \frac{\partial c_k}{\partial \theta}, \quad (15)$$

where θ is, as usual, a parameter.

For all three cases, the gradient updates depend on the p_i values, and we use standard backpropagation along the computation graph.

	Perplexity	ROUGE2
GRU	123.0	12.1
MuFuRu	119.7	14.7
Our model	101.4	17.2

Table 1: Experiment Results of Language Modeling.

4 Empirical Evaluation

We evaluate our model on a range of different sequential learning tasks, in order to demonstrate the improvements obtained with our enhanced model.

4.1 Language Modeling

We use the standard Penn Treebank dataset [Marcus *et al.*, 1993] for language modeling. To measure the effectiveness of our language models, we consider the standard perplexity and ROUGE2 measures. Recall that perplexity is defined as

$$\text{Perplexity} = \exp\left(\frac{1}{N}\sum_{i=1}^{N} -\log p(w_i \mid \cdots, w_{i-1})\right), \quad (16)$$

where w_i is the i-th word and N is the length of the sentence. ROUGE2 is defined as the recall over bigrams. We compare our model with two strong baselines, GRUs and MuFuRu [Weissenborn and Rocktäschel, 2016]. At test time, we use beam search to find the best word sequence. The results are given in Table 1. Compared against the baselines, our model achieves the best performance both in terms of perplexity and in terms of the ROUGE2 measure. Our model reduces the perplexity of GRUs by 21.3% and that of MuFuRUs by 18.0%. At the same time, our model improves by 42% over GRUs and by 17% over MuFuRUs in terms of the respective ROUGE2 scores.

Additionally, we evaluate our model on two further datasets: (1) Restaurant Reservation Dialogs [Eric and Manning, 2017] and (2) the Switchboard Dialog Act Corpus [Liu and Lane, 2017]. The first dataset consists of raw dialogue text and system commands. The dialogue contains both user and system utterances. The system commands are used to query its knowledge base in order to respond to a users' utterance. The average number of utterances in each dialogue is 14. The vocabulary size is 1,229. The dataset is split into training set, test set, and validation set, with 1,618, 1,117, and 500 dialogues, respectively. In the second (Switchboard) dataset, there are part-of-speech (POS) tags associated with the utterances. The data is split into 14 equal parts, 10 of which are used for training, one for testing, and the remaining three for validation. In total, there are 223.5K utterances in the corpus. Only the most frequent 10K words are considered for the vocabulary.

For the restaurant reservation dialog dataset, the previous state-of-the-art [Eric and Manning, 2017] achieves an accuracy of 48%, while our result is 51.3%. Our model improves the accuracy of the previous state-of-the-art by 6.8%. The performance comparison for the Switchboard Dialog Act Corpus is shown in Table 2. Our model reduces the perplexity of the previous state-of-the-art [Liu and Lane, 2017] by 8.8%.

	Perplexity
Previous state-of-the-art	58.4
Our result	53.7

Table 2: Experiment Results of Dialogue Corpora.

Algorithms	Accuracy
Attentive reader	0.53
Stanford reader	0.64
AS reader	0.57
GA reader	0.57
Our model	0.69

Table 3: Experiment Results of Cloze tests.

4.2 Cloze Test

In a Cloze test, one is given a sentence, from which one of the words has been excised and left blank. The goal is to fill the blank with an appropriate word, choosing among different options. We use the "Who-did-What" dataset [Onishi *et al.*, 2016], which consists of around 0.2 million cloze questions. On average, there are 3.5 different choices for each blank. The total vocabulary size is more than 0.3 million. We compare with four neural network baselines:

1. Attentive Reader [Hermann *et al.*, 2015]: LSTM with attention mechanism.
2. Stanford Reader [Chen *et al.*, 2016]: As above, but with a bilinear term.
3. Attention Sum (AS) Reader [Kadlec *et al.*, 2016]: GRU with attention.
4. Gated-Attention (GA) Reader [Dhingra *et al.*, 2016]: As above, but with gated layers.

The accuracy scores of these baselines and our model are shown in Table 3. It can be seen that our model is 5% better than the best baseline on the "Who-did-What" dataset.

4.3 Character-Level Word Embedding

In this subsection, we evaluate on the character-to-word embedding task, a supervised task, where the goal is to predict a word embedding given a sequence of character embeddings. There are several challenges. First, each word has a different number of characters. This is naturally addressed by using the RNN structure, which is well-suited for variable-length inputs. Second, the dimensions of the character embedding and the word embedding diverge significantly (the former is around 10, while the latter is around 400). For the second problem, we use a non-square matrix to transform the input vector to the state vector, and another non-square matrix to transform the state vector to the output vector.

The intuition for why our model may work better for this character-to-word task than previous methods is that the data may be bimodal in the sense that there are two kinds of words. As a concrete example, consider the words "inconsiderable" and "piano". While the semantics of the former is reasonably predictable from the standard morphological rules of the English language, the latter is borrowed from a foreign language. This shows that there are at least two logics inherent to the data, which can easily be captured by our multi-weight model, but is challenging for previous methods.

We compare our model with the model of Santos et al. [2014] and use the Penn Treebank dataset (standard Wall Street Journal portion) [Marcus *et al.*, 1993]. The ground truth word embedding is generated using the standard *word2vec* tool using SGNS [Mikolov *et al.*, 2013]. The accuracies of Santos et al. and of our model are 97.3% and 98.1%, respectively. Again, we find that our model outperforms previous work; in this case, the method of Santos et al. by 0.8% (absolute).

4.4 Analysis and Discussion

We next shed further light on why and how our multi-weight approach succeeds, while also providing an error analysis.

Theoretical Analysis. For simplicity, we assume there is a mixing of two separated tasks and set the number of matrices to two. The case of more than two matrices is analogous. In particular, we consider the instructive example of interleaving English speech generation and French speech generation, as is common in code-switching and in multilingual announcements.

Initially, the parameters in our model will be quite random. During training, the weight matrices and state vectors will stabilize to a steady state. In this steady state, ideally, each character within an English utterance would result in selecting the weight matrix W_1, while each character within a French utterance would select for the weight matrix W_2. In this case, our model can be understood as a seamless combination of two language models, the English model and the French model. This explains our model's ability to deal with data in which different aspects are blended together, and its ability to switch between them.

The reader may have noticed that similar mechanisms also appear in a different form in attention models. However, there are a few notable differences. First, in an attention model, a weight is used to determine to what extent to consider the hidden state obtained for a given input, e.g. different keywords in a sentence [Wang *et al.*, 2016], while in our multi-weight LSTM model, the focus is instead on choosing the right weight matrices to produce a hidden state. Second, for attention models, the steps that require attention are usually fixed, while in our model, the number of selections may vary for different instances. Third, attention mechanisms typically are used to enable each output step to focus on the hidden states for different parts of an input sequence. Our gating mechanism instead enables entirely different *operations*, not just varying attention, and enables this for every *input* step, while an attention model applies only after the entire input has been consumed. Fourth, unlike attention mechanisms, the weight matrices for these different operations are automatically learned from the data.

Experimental Analysis. In order to validate these theoretical intuitions experimentally, we create two artificial datasets. The first of these consists of the union of a large biomedical corpus [Bunescu and Mooney, 2005] and a news corpus [Das

Bio	News
0.81	0.27
English	French
0.15	0.93

Table 4: The relation between the selected weight matrix and the type of sentence.

Error class	Percentage
Infrequent Mechanism	63%
Prior Knowledge	21%
Others	16%

Table 5: Distribution of different error classes.

and Bandyopadhyay, 2010], with a randomly shuffled sentence order. In the second dataset, we take an English corpus [Marcus *et al.*, 1993] and another French corpus [Degand and Bestgen, 2003], and again randomly shuffle the sentences.

Now, we examine and interpret the cell behavior on these manufactured datasets. In our experiment, we again set the number of weight matrices to be two, since it suffices for capturing the bimodal nature of these datasets. After training our model and before prediction, we keep track of which corpus each test sentence originated from and trace the value of p_i in Eq. (8) of the cell. Recall that p_i denotes the probability of selecting the i-th weight matrix W_i.

In Table 4, we list the corresponding p_1 for the respective two corpora in the two datasets. Since, in this two-weight matrix instantiation of multi-weight LSTMs, $p_2 = 1 - p_1$ can be determined from p_1, we omit its values in the table. It can be seen that, in both datasets, one kind of data selects W_1 much more often, while the other goes through W_2 much more often.

Error Analysis. To further improve the architecture of our framework in the future, it is helpful to scrutinize the failure cases of our models. In particular, we have categorized the failure cases observed for the cloze tests into the following categories.

1. Infrequent Mechanism. For example, the choices for the blank in "... *Sources close to the presidential palace said that Fujimori declined at the last moment to leave the country and instead he will send a high level delegation to the ceremony, at which Chilean President Eduardo Frei will pass the mandate to _____. ...*" are (1) Augusto Pinochet, (2) Jack Straw, and (3) Ricardo Lagos. Our model wrongly chooses option (2), while the correct answer is choice (1). The model may have misunderstood that Eduardo Frei is in close contact with Jack Straw, since the name Jack Straw is closest to Eduardo Frei among the three choices. Such errors likely occur because this is an infrequent case not covered by the main mechanisms of inference learnt by our model.
2. Prior Knowledge: For example, the choices for the blank in "... *U.S. officials, repeatedly stymied in the quest to oust alleged war criminals from the Bosnian Serb leadership, said Tuesday that increased NATO patrols could improve the chances that indicted suspects such as Radovan Karadzic and _____ will be arrested. ...*" are (1) Warren Christopher, (2) Nicholas Burns, (3) George Joulwan, (4) Simon Haselock, and (5) Ratko Mladic. Our model wrongly chooses option (3), while the correct answer is choice (5). It is not mentioned that Ratko Mladic is a criminal in the context, though it is common knowledge, and hence the model may not have captured that the word "suspect" hints at the blank being filled with this name. Overcoming such errors may require an additional dictionary or encyclopedia.

The relative percentages of different classes of errors are given in Table 5. For this evaluation, we randomly select 100 failure cases and categorize them manually. It can be seen that the class of Infrequent Mechanism errors occupies the largest portion, followed by Prior Knowledge, which comes second. Thus, focusing on avoiding these two types of errors may greatly enhance the success rate of our model on cloze test scores specifically and other tasks in general.

5 Conclusion

In summary, we have proposed a novel framework for multi-weight RNN cells that rely on multiple weight matrices to better cope with multi-faceted data. The mechanism is explained theoretically and experimentally analyzed in substantial detail. Extensive experiments on multiple tasks show the consistent superiority of our model against previous state-of-the-art models. Since most real-world data is multi-faceted in nature, we expect this approach to be of broad applicability in numerous different tasks.

References

[Bengio and others, 2009] Yoshua Bengio et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[Bengio *et al.*, 1994] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[Bunescu and Mooney, 2005] Razvan C. Bunescu and Raymond J. Mooney. Subsequence kernels for relation extraction. *Advances in Neural Information Processing Systems*, pages 171–178, 2005.

[Caruana, 1997] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[Chen *et al.*, 2016] Danqi Chen, Jason Bolton, and Christopher D Manning. A thorough examination of the cnn/daily mail reading comprehension task. *arXiv preprint arXiv:1606.02858*, 2016.

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

[Chung *et al.*, 2014] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[Das and Bandyopadhyay, 2010] D. Das and S. Bandyopadhyay. Sentence level emotion tagging on blog and news corpora : Journal of intelligent systems. *Journal of Intelligent Systems*, 19(2):145–162, 2010.

[Degand and Bestgen, 2003] Liesbeth Degand and Yves Bestgen. Towards automatic retrieval of idioms in french newspaper corpora. *Literary & Linguistic Computing*, 18:249–259, 2003.

[Dhingra *et al.*, 2016] Bhuwan Dhingra, Hanxiao Liu, William W Cohen, and Ruslan Salakhutdinov. Gated-attention readers for text comprehension. *arXiv preprint arXiv:1606.01549*, 2016.

[Eric and Manning, 2017] Mihail Eric and Christopher D Manning. A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. *arXiv preprint arXiv:1701.04024*, 2017.

[Fausett, 1994] Laurene V Fausett. *Fundamentals of neural networks*. Prentice-Hall, 1994.

[Glorot *et al.*, 2011] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 513–520, 2011.

[Greff *et al.*, 2016] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 2016.

[Hermann *et al.*, 2015] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015.

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

[Jozefowicz *et al.*, 2015] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 2342–2350. JMLR.org, 2015.

[Kadlec *et al.*, 2016] Rudolf Kadlec, Martin Schmid, Ondrej Bajgar, and Jan Kleindienst. Text understanding with the attention sum reader network. *arXiv preprint arXiv:1603.01547*, 2016.

[Le *et al.*, 2015] Quoc V Le, Navdeep Jaitly, and Geoffrey E Hinton. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*, 2015.

[Liu and Lane, 2017] Bing Liu and Ian Lane. Dialog context language modeling with recurrent neural networks. *arXiv preprint arXiv:1701.04056*, 2017.

[Marcus *et al.*, 1993] Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.

[Mikolov and Zweig, 2012] Tomas Mikolov and Geoffrey Zweig. Context dependent recurrent neural network language model. *SLT*, 12:234–239, 2012.

[Mikolov *et al.*, 2010] Tomas Mikolov, Martin Karafit, Lukas Burget, Jan Cernock, and Sanjeev Khudanpur. Recurrent neural network based language model. In *INTERSPEECH 2010, Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September*, pages 1045–1048, 2010.

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

[Mikolov *et al.*, 2014] Tomas Mikolov, Armand Joulin, Sumit Chopra, Michael Mathieu, and Marc'Aurelio Ranzato. Learning longer memory in recurrent neural networks. *arXiv preprint arXiv:1412.7753*, 2014.

[Onishi *et al.*, 2016] Takeshi Onishi, Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. Who did what: A large-scale person-centered cloze dataset. *arXiv preprint arXiv:1608.05457*, 2016.

[Santos and Zadrozny, 2014] Cicero Nogueira Dos Santos and Bianca Zadrozny. Learning character-level representations for part-of-speech tagging. In *International Conference on Machine Learning*, pages 1818–1826, 2014.

[Wang and Cho, 2015] Tian Wang and Kyunghyun Cho. Larger-context language modelling. *arXiv preprint arXiv:1511.03729*, 2015.

[Wang *et al.*, 2016] Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. Relation classification via multi-level attention CNNs. In *Proceedings of ACL 2016*, 2016.

[Weissenborn and Rocktäschel, 2016] Dirk Weissenborn and Tim Rocktäschel. Mufuru: The multi-function recurrent unit. *arXiv preprint arXiv:1606.03002*, 2016.

[Zeiler, 2012] Matthew D. Zeiler. Adadelta: An adaptive learning rate method. *Computer Science*, 2012.

[Zhang *et al.*, 2016] Dong Zhang, Shoushan Li, Hongling Wang, and Guodong Zhou. User classification with multiple textual perspectives. In *Proceedings of COLING*, pages 2112–2121, 2016.

[Zhou *et al.*, 2016] Guo-Bing Zhou, Jianxin Wu, Chen-Lin Zhang, and Zhi-Hua Zhou. Minimal gated unit for recurrent neural networks. *International Journal of Automation and Computing*, 13(3):226–234, 2016.