A Review on Neural Language Modeling

Anuar Maratkhan

School of Science and Technology
Nazarbayev University
anuar.maratkhan@nu.edu.kz

***Abstract*—Language modeling (LM) appears to be a main task of natural language processing (NLP) due to use in speech recognition, machine translation, optical character recognition. Language modeling is also known as statistical language modeling due to the large number of statistical approaches. However, in the last decade neural language models became more popular due to higher performance results in terms of perplexity. Moreover, recurrent neural network (RNN) model appears to be beneficial for neural language modeling, especially those that are optimized with backpropagation. The study will therefore, review some of the previous and current state-of-the-art neural language models, and present different studies involving word-level and subword-level recurrent neural network approaches, including those obtained with reinforcement learning method. The study then ends by summing up those discussions and giving some future work suggestions.**

I. Introduction

Language Modeling (LM) is a central task in Natural Language Processing (NLP) and play main role in speech recognition, machine translation, optical character recognition, natural language understanding, question answering and many other tasks. Language modeling is all about sequential data. *Language model* is an algorithm for predicting next word in a text given preceeding ones (figure 1). The figure given below is a good demonstration of how language model works. The model is given a context ”I have a pen. I have an”, and then is required to predict the consequent word. The example figure shows that model chooses from words ”apple”, ”pen”, ”red”, and ”hello” as a possible next word. Further, according to the probability assigned to each of those words, the model outputs one word for modeling the language. In addition, the model can treat the words in the context as whole (as it can be seen from figure 1), or the model also can treat the words on the character-level and subword-level.

It is also mostly known as *statistical language model* for its richness of statistical approaches in previous works. Statistical language modeling starts by forming document-term matrix which displays the occurence frequency of each word in a text, and then based on that the model measures the probability of occurence of particular word in sequence. The state-of-the-art statistical language models turned up to be cache models and class-based models. In addition, according to [1], most of the statistical models require more data for better performance. Because of that, statistical language models lose to neural language models because neural models require less training data compared to statistical models.

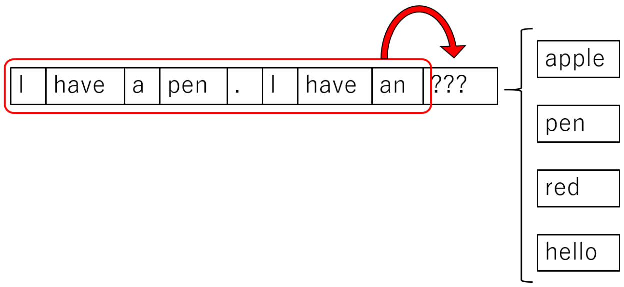

Fig. 1. Next word prediction in a text.

Later, as soon as neural networks became popular (due to invention of optimization methods), *neural language modeling* approaches were invented. Artificial neural networks perform well on supervised learning tasks (when the data is *labeled*) and unsupervised learning tasks (when the data is *unlabeled*). Quoc V. Le et al. in their study [2] explored and demonstrated how well neural networks operate on unsupervised learning task, where a model has to detect whether a given image contains a face or not. And as the study [2] notes, the work has been motivated by neuroscience hypothesis. So does the whole science about neural networks. Moreover, information retrieval tasks that are mainly composed of unlabeled data, like in search engines, can be approached by such unsupervised learning techniques. Neural language models, also known as continous-space language models, make use of these neural networks. Those are current state-of-the-art approaches in language modeling.

In the second section of this paper we start by defining what the Recurrent Neural Network (RNN) is, and how it can be applied to language modeling. Studies [1], [3], [4], [5] show that RNNs seem to be the best techniques for language modeling in terms of performance. Thus, RNN appears to be a main approach for neural language modeling nowadays but not the only one. Hence, we proceed by reviewing other interesting approaches to neural language modeling such like subword-level language models presented in [6] and reinforcement learning based language models like in [7]. We continue by concluding the review of artificial neural science applied in language modeling, and present our future suggestions on the neural language modeling approaches.

II. Recurrent Neural Networks based Language Models

Even though recurrent neural networks were invented relatively long ago, researchers started to use reccurent neural networks for language modeling tasks in the last decade.

Recurrent neural networks based language models produce state-of-the-art results nowadays in terms of performance. Hence, this section of the study will define and describe what recurrent neural network is, and how based on those neural networks language models perform.

A. Recurrent Neural Network

Recurrent Neural Networks (RNN) are primarily used in sequential data analysis such as video processing, text processing, predicting stock prices and other. RNN architecture is an improvement of feed-forward neural networks that takes into account previous hidden layer results by storing them in memory. An invention of backpropogation caused an exciting use of RNNs [8].

In the last works [1], [3], [4] RNNs proved that they are perfect for language modeling tasks. In particular, simple recurrent neural network used in [1] can be described by input at time t – $x(t)$, hidden layer at time t – $s(t)$, and the output layer at time t – $y(t)$, which are computed as follows:

$$x(t) = w(t) + s(t-1) \tag{1}$$

$$s_j(t) = f\left(\sum_i x_i(t) u_j i\right) \tag{2}$$

$$y_k(t) = g\left(\sum_j s_j(t) v_k j\right) \tag{3}$$

where (1) uses current word at time t and previous hidden layer at time $t-1$ as input, (2) and (3) have weights u and v used for computations, and both also have sigmoid activation function and softmax function, which are:

$$f(z) = \frac{1}{1+e^{-z}} \tag{4}$$

and

$$g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}} \tag{5}$$

respectively.

The problem of simple RNN is that the backpropogated gradients vanish or explode [9]. That means that simple recurrent neural network model can not store long context but just several preceeding words. Due to the short-term memory of such simple neural network architecture in RNN, the efficiency to predict next word in the sequence lowers. Therefore, an improvement in RNN architecture further was introduced with invention of Long-Short Term Memory (LSTM). The LSTM solved the vanishing gradient problem by having special cells with activation gates that provided the model to store longer context. Moreover, Gated Recurrent Unit (GRU), neural network architecture, improved the architecture of the LSTM by combining two gates that made the model simpler but without loss in performance. Overall, the study [9] shows that RNNs perform better than complex statistical approaches, N-grams, on a real-world data. Thus, recurrent neural network architecture is the best model to use in the language modeling tasks in terms of the performance.

B. Language Models

Neural language models have demonstrated relatively superior performance on language modeling and speech recognition tasks in comparison to the state-of-the-art statistical approaches, class-based models, in the last decades. Neural language models take their first steps from taking the advantage of feed-forward network based models presented in the works listed in [1], which motivated T. Mikolov et al. to investigate the performance of more advanced type of neural network, RNN. The study was first to evaluate RNN for modeling such sequential data, and as a result showed state-of-the-art performance.

In contrast to feed-forward networks that use fixed size of context to predict next word in a sequence, RNNs does not require to use limited number of preceeding words [1]. Hence, RNNs have dynamic size of context in the memory that enables to store unlimited number of preceeding words for better prediction in a sequence. The basic structure of simple recurrent neural network used in the study is shown in figure 2 below. In the figure, the flow of whole process of prediction is shown from left to right. First, the input at a particular time comes as word, and then it flows to the hidden layer which at the same time receives previous hidden layer result as context. The generated hidden layer later will pass the result to the next propagation through the network at time $t+1$, which will form larger context. Soon after the long context is formed in hidden layer, the prediction of the next word is outputted.

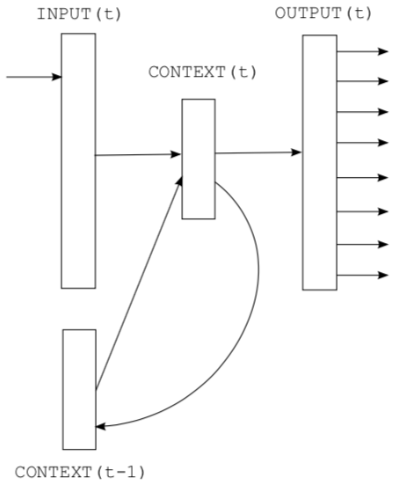

Fig. 2. Simple recurrent neural network from [1]

According to [1] study results RNNs illustrated excellent performance relative to the best backoff models even being trained on less data than the second ones. The performance in language modeling is measured by *perplexity* which illustrates how well the model can predict next word in a sequence. The lower the perplexity is, the higher the performance. On the other hand, the higher the perplexity, the lower the performance of a particular language model. Therefore, the study shows much lower perplexity compared to statistical language modeling approaches.

Furthermore, [3] work presents that RNN based LM can achieve higher efficiency on language modeling tasks by being trained with backpropogation through time (BPTT) optimization technique, which serves as an extension to standard backpropogation algorithm. This backpropagation algorithm is main optimization technique of neural network models. The backpropagation algorithm is basically updating neural network weights (coefficients) iteratively. The key concept of BPTT is that the error propogates back for several time steps through its recurrent connections, as it was stated in [3].

Later, Zaremba et al. were first to implement LSTM model with dropout regularization for language modeling task, and demonstrated another improvements in perplexity [4]. Moreover, subsequent studies listed in [5] show other improvements in perplexity with LSTMs and other techniques on word-level language modeling. Short overview on the work done there is illustrated in table I[1]. Furthermore, it can be clearly seen from table I that from the beginning of neural language modeling, the perplexity results of neural models reduced significantly (from 92.0 down to 47.69 presented in [5]). Number of parameters approximately stays the same, except [1] work. In addition, LSTM models appear to be the most popular in neural language modeling. Overall, it is also clear that the performance of those models from table I advanced a lot in short period of time starting from 2012 [1] till 2017 [5], and will continue to enhance.

In general, recurrent neural networks are expected to improve natural language understanding in real-world applications. These applications also involve language modeling tasks. The studies reviewed above prove the benefits of using RNN architecture in language modeling which perform better than the best backoff models. The RNN approach to language modeling lowers the perplexity, and lowers the size of dataset for training the model. In addition, as it was stated, RNNs have unlimited size of context which produce better prediction of consequent words. Moreover, the discussed BPTT optimization and dropout regularization techniques enhances the performance of RNN more than those without usage of such optimization and regularization methods.

III. Other Language Models

Although the reccurent neural architecture is presented as the most intelligent approach for language modeling, the basic RNN way of modeling the language is not the only one. There are dozen of other language models presented. This section reviews RNN architecture used for subword-level language modeling [6], and neural architecture search for language modeling presented in [7].

A. Subword-level Language Modeling

All of the studies discussed before involved word-level language modeling approaches. Word-level language models have particular size of vocabulary that indicates number of *unique* words learned in the process of training. Even though those models show relatively superior performance in terms of lower perplexity and lower number of parameters, when it comes to new words (during evaluation on the test set) that have not appeared throughout training, the word-level language models are unable to assign nonzero probability of occurence to those unknown Out-of-Vocabulary (OOV) words. As a consequence, the models are incapable to use OOV words to predict next word in the sequence. If the OOV words are rare words that occur in the texts not so often, it is not that dangerous because chances of occurrence of those infrequent words are low. But when the OOV words occur not rarely but often, OOV words become a significant problem to word-level language models. To eliminate this problem, statistical language models have been using *character-level* language modeling techniques, where model trains not on the words in the document but on the individual characters in the words. From those statistical approaches to character-level modeling, neural language models that involved the same character-level modeling emerged.

According to [6] study, a different approach to language modeling tasks, character-level modeling, showed lower results in comparison to word-level language models that engaged simple feed-forward neural network, recurrent neural network, n-discounted n-gram, and maximum entropy n-gram on Penn Treebank and text8 datasets. Besides that, however, because new words are independent of vocabulary in the character-level models, the results have shown zero OOV. Moreover, the study [6] states that the character-level models are hardly trained in order to show high performance, and still even neural network architectures with 1000 neurons on hidden layer (comparatively large hidden layer) are not as good as word-level language models. Therefore, T.Mikolov et al. suggests to use different approach to language modeling, the subword-level model, that can be as efficient as word-level model and as general as character-level language models. The example of conversion of words into subwords is given in figure III-A below. First line in figure III-A indicates the approach of using word-level language model where words as whole are passed as input. Second line shows how subword-level language model processes newly occured words in the text by partitioning the OOV word into subwords (colon character ":" signifies the end of word).

```
new company dreamworks interactive
new company dre+ am+ wo+ rks:  in+ te+ ra+ cti+ ve:
```

Furthermore, the [6] study shows the improvements in performance of language models using subword-level approach instead of character-level. In addition, subword-level language modeling reduces the number of parameters in neural network architecture, which is computed as follows:

$$\#ofparameters = (2 \times V + H) \times H \tag{6}$$

where V – the size of vocabulary, and H – the size of hidden layer.

Thus, the subword-level language modeling is worth considering for usage in language modeling tasks due to its combination of advantages of both word-level and character-level

[1] It is out of scope of this paper to provide detailed description of those approaches.

TABLE I
SINGLE MODEL PERPLEXITY ON VALIDATION AND TEST SETS ON PENN TREEBANK [5].

Model	**Param**	**Validation**	**Test**
Mikolov & Zweig (2012) – RNN-LDA + KN-5 + cache	9M	-	92.0
Zaremba et al. (2014) – LSTM	20M	86.2	82.7
Gal & Ghahramani (2016) – Variational LSTM (MC)	20M	-	78.6
Kim et al. (2016) – CharCNN	19M	-	78.9
Merity et al. (2016) – Pointer Sentinel-LSTM	21M	75.7	70.9
Grave et al. (2016) – LSTM + continuous cache pointer	-	-	72.1
Inan et al. (2016) – Tied Variational LSTM + augmenter loss	24M	75.7	73.2
Zilly et al. (2016) – Variational RHN	23M	67.9	65.4
Zoph & Le (2016) – NAS Cell	25M	-	64.0
Melis et al. (2017) – 2-layer skip connection LSTM	24M	60.9	58.3
Merity et al. (2017) – AWD-LSTM w/o finetune	24M	60.7	58.8
Merity et al. (2017) – AWD-LSTM	24M	60.0	57.3
Salakhutdinov et al. [5] – AWD-LSTM-MoS w/o finetune	22M	58.08	55.97
Salakhutdinov et al. [5] – AWD-LSTM-MoS	22M	**56.54**	**54.4**
Merity et al. (2017) – AWD-LSTM + continous cache pointer	24M	53.9	52.8
Krause et al. (2017) – AWD-LSTM + dynamic evaluation	24M	51.6	51.1
Salakhutdinov et al. [5] – AWD-LSTM-MoS + dynamic evaluation	22M	**48.33**	**47.69**

TABLE II
SINGLE MODEL PERPLEXITY ON THE TEST SET OF THE PENN TREEBANK LANGUAGE MODELING TASK [7].

Model	**Parameters**	**Test Perplexity**
Neural Architecture Search with base 8	32M	67.9
Neural Architecture Search with base 8 and shared embeddings	25M	64.0
Neural Architecture Search with base 8 and shared embeddings	54M	62.4

approaches. The benefits of subword-level language modeling approach are, therefore, low number of parameters used by neural network and handling of OOV words in the document.

B. Reinforcement Learning

Reinforcement learning is an area of deep learning with a specific agent and an environment where the agent is rewarded according to its performance on some task. For instance, B.Zaph and Quoc V. Le in their study [7] present a novel reinforcement learning algorithm that searches best neural network architecture for performing language modeling tasks (refer to figure 3).

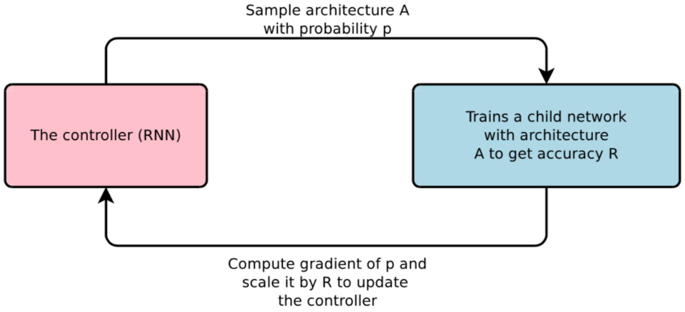

Fig. 3. An overview of Neural Architecture Search from [7]

As it can be seen from figure 3, the agent gives reward signal to the controller that tries different neural network architectures based on the accuracy result of the neural network. As a result, the controller produces a neural network architecture that achieves higher accuracy performance further. For better understanding, one may imagine a controller parent, recurrent neural network, that produce child, a new neural network architecture, and gets rewarded on the performance of the child. It is the same as if the hardware engineer (parent) develops a robot (child), and based on the performance of robot on a given task, the engineer gets rewarded.

The algorithm in [7] constructs a tree of architectures which RNN controller then uses. The search space for the controller includes 1) a combination method – *addition*, *multiplication*, and 2) an activation function – *identity*, *tanh*, *sigmoid*, *relu* (refer to figure **??**). The authors of [7] evaluated the performance of their neural architecture searcher on a well-known dataset for language modeling, Penn Treebank. The table II demonstrates the obtained results of [7] work.

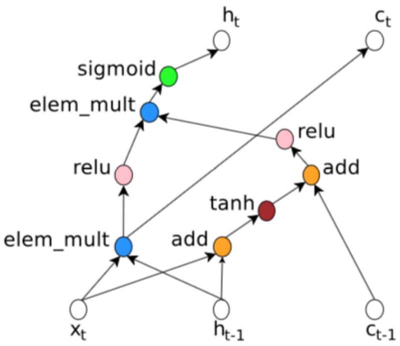

Fig. 4. The computation graph of the recurrent cell constructed from example predictions of the controller in [7]

Hence, the reinforcement learning techniques also can support language models with a guidance on how to achieve better accuracy results. The reviewed study on reinforcement learning illustrate how well the RNN controller can learn to produce and find the most accurate model. As it can be seen from table II, the neural network with 8 layers in the encoder, and 8 layers in the decoder produce the perplexity results that outperform some of the previous state-of-the-art approaches shown in table I to language modeling with equally the same amount of parameters (25 million).

IV. Conclusion and future work

Language modeling appeared to be a main task in a variety of NLP tasks. Therefore, in the previous century statistical language models were invented to make prediction of the next word in a document easier using statistics. However, in the last decades neural language models or also called continous-space language models have started to be used.

Over the past few years, it was clearly seen that continous-space language models outperformed statistical modeling techniques significantly. The ease of feature learning without need in extraction of those features simplified lives of the scientists. The neural approach offers such an opportunity to its users.

There was a number of studies showing that neural language models outperforms statistical models in terms of perplexity and in addition do not require large datasets as backoff models. This leads us to the fact that neural language models are current state-of-the-art techniques in language modeling. Those neural language approaches include feed-forward neural networks, simple recurrent neural networks, improved recurrent neural networks (LSTM) on a word-level and subword-level obtained in the standard and non-standard (reinforcement learning) ways.

As we have already observed, there are lots of different neural solutions to language modeling tasks. And as we have seen in [7], combining several techniques results in newer improvements of existing models. So, the study suggests to combine different approaches in the way that will obtain better results in perplexity. For instance, one may combine the subword-level RNN (LSTM) based language model with deep reinforcement learning algorithm.

References

[1] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, “Recurrent neural network based language model,” in *Interspeech*, vol. 2, 2010.

[2] Q. V. Le, M. A. Ranzato, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Deau, and A. Y. Ng., “Building high-level features using large scale unsupervised learning,” in *Proceedings of the 29 th International Conference on Machine Learning*, Edinburgh, Scotland, UK, 2012.

[3] T. Mikolov, S. Kombrink, L. Burget, J. Cernocký, and S. Khudanpur, “Extensions of recurrent neural network language model,” *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5528–5531, 2011.

[4] W. Zaremba, I. Sutskever, , and O. Vinyals, “Recurrent neural network regularization,” 2014, arXiv preprint arXiv:1409.2329.

[5] Z. Yang, Z. Dai, R. Salakhutdinov, and W. W. Cohen, “Breaking the softmax bottleneck: A high-rank RNN language model,” 2017.

[6] T. Mikolov, I. Sutskever, A. Deoras, H.-S. Le, S. Kombrink, and J. Cernocký, “Subword language modeling with neural networks,” 2011.

[7] B. Zoph and Q. V. Le, “Neural architecture search with reinforcement learning,” 2017, arXiv preprint arXiv:1611.01578.

[8] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Macmillan Publishers Limited*, vol. 521, pp. 436–444, 2015.

[9] A. Karpathy, J. Johnson, and L. Fei-Fei, “Visualizing and understanding reccurent networks,” 2015, under review as a conference paper at ICLR 2016.