TDTO Language Modeling with Feedforward Neural Networks

Tze Yuang Chong1,2,3, Rafael E. Banchs3, Eng Siong Chng1,2, Haizhou Li1,2,3

1 Temasek Laboratories, Nanyang Technological University, Singapore 639798
2 School of Computer Engineering, Nanyang Technological University, Singapore 639798
3 Institute for Infocomm Research, Singapore 138632
tychong@ntu.edu.sg, rembanchs@i2r.a-star.edu.sg,
aseschng@ntu.edu.sg, hli@i2r.a-star.edu.sg

Abstract

In this paper, we describe the use of feedforward neural networks to improve the term-distance term-occurrence (TDTO) language model, previously proposed in [1]−[3]. The main idea behind the TDTO model proposition is to model separately both position and occurrence information of words in the history-context to better estimate n-gram probabilities. Neural networks have been shown to offer a better generalization property than other conventional smoothing methods. We take advantage of such property for a better smoothing mechanism for the TDTO model, referred to as the continuous space TDTO (cTDTO). The newly proposed model has reported an improved perplexity over the baseline TDTO model of up to 9.2%, at history length of ten, as evaluated on the Wall Street Journal (WSJ) corpus. Also, in the Aurora-4 speech recognition N-best re-ranking task, the cTDTO outperformed the TDTO model by reducing the word error rate (WER) up to 12.9% relatively.

Index Terms: term-distance, term-occurrence, TDTO, continuous space, language model, speech recognition

1. Introduction

Recently, we have suggested a new language modeling approach, in which information related to position\(^1\) and occurrence\(^2\) of words in the history-context, is decoupled from each other [1]−[3]. In the proposed model, coined as the term-distance term-occurrence (TDTO) model, the position and the occurrence events are captured from the history-context by using two different model components. By fusing these components appropriately, the contribution of the position and the occurrence information towards the final prediction can be explicitly computed. Empirical results showed that different history lengths, positions and occurrences of history-words contribute differently; e.g. for a short history-context, word position has a higher impact but as the history-context grows, word occurrence becomes more dominant.

Despite exhibiting promising results, smoothing remains a critical issue to the TDTO model. Smoothing is important for estimating more accurate probabilities, in particularly for the rare and unseen events. In our previous works, the probability distributions of position and occurrence were smoothed by using a running average filter and a back-off mechanism, respectively [1]−[3]. Also, the interdependences among history-words were assumed conditional independence. However, the resulted TDTO model might not still generalize well.

In this paper, we focus on the problem of smoothing the TDTO model. Inspired by the observed advantages of neural networks in modeling n-grams [4]−[18], we explore the use of neural networks to learn word probabilities based on the original proposition of the TDTO model [1]−[3]. The idea is to project both events of position and occurrence into continuous space from which smoother probability distributions can be estimated. As similar events are expected to be represented closely in the continuous space, the probabilities of the rare and unseen events can be interpolated more effectively.

Next section provides some background information about the TDTO model. Section 3 presents our proposed continuous space version of the TDTO model, referred to as the cTDTO model, which is implemented by using feedforward neural networks. Section 4 discusses the experimental results and Section 5 presents our conclusion.

2. TDTO language model

The aim of the TDTO model is to exploit longer history-contexts more effectively for language modeling purposes. Several different approaches have been proposed to this end. To reduce the complexity of modeling long history-context, these approaches either ignore the position information in the history-context [20]−[24], or assume the occurrences of certain history-words to be independent from the target-word [25]−[28].

The TDTO model takes a different approach to handle the complexity resulting from modeling long contexts. First, the presence of a given word in the history-context is decoupled into two events: position and occurrence. The occurrence events carry the coarse information of the existence of the words in the history-context while, the position events give details on how these words are arranged in the history-context. According to this, the TDTO model utilizes two model components to capture these two types of events. By appropriately weighting the contribution of each component, we optimize the performance of the overall model according to the target applications.

Consider the history-context \(h \) of length \( n-1 \), say \( w_{r-1}^{r-1} \), that precedes a target-word \( t \), say \( w_r \), the history-context \( h \) can be expressed in terms of positions and occurrences of the history-words as follows.
where $\Delta(w_k)$ denotes the position of history-word $w_k$ as measured from the target-word $w_i$, i.e. the term-distance (TD) event; while $w_k \in h$ denotes the occurrence of word $w_k$ within the history-context of the target-word $w_i$, i.e. the term-occurrence (TO) event.

In our previous implementation [1]−[3], we disregarded the interdependences among the TD and TO events among the different words composing the history-context. Such simplification allows for the TD and the TO probabilities to be feasibly computed by considering the distance and the co-occurrence of the word-pairs of target and history words.

The final formulation of the TDTO modeling is as follows:

$$h = \bigcap_{k=i-1}^{i+n-1} \Delta(w_k) = k \cap \{w_k \in h\}$$ (1)

$P(t = w_i|h = w_i^{-1}w_{i+1})$

$$P(t = w_i|h_N = w_i^{-1}w_{i+1})^{\beta_H}$$

$$\prod_{k=1}^{i-n-1} P(\Delta(w_k) = k|w_k \in h, t = w_i)^{\beta_D}$$

$$\approx \frac{\prod_{k=1}^{i-n-1} P(w_k \in h|t = w_i)^{\beta_O}}{Z(h)}$$ (2)

where $\Delta(w_k)$ denotes the position of the considered history-word $w_k$ (its distance from the target-word $w_i$) and $w_k \in h$ denotes the occurrence of word $w_k$ in the history of word $w_i$.

As seen from (2), the TDTO model is composed of three model components:

- An $n$-gram prior component that considers a short history-context $h_N = w_i^{-1}w_{i+n}$, where $n_N = 3 < n$.
- A term-distance (TD) likelihood that measures how likely distance $k$ would be encountered between history-word $w_k$ and target-word $w_i$.
- A term-occurrence (TO) likelihood that measures how likely history-word $w_k$ would be seen in the history-context of target-word $w_i$.

These components are weighted by $\beta_N$, $\beta_D$, and $\beta_O$ accordingly.

3. **Language model smoothing by using feedforward neural networks**

Various smoothing techniques have been proposed for the $n$-gram language model (see [29] for a thorough review). Basically, these techniques are based on redistributing the observed count mass among the discrete classes of $n$-grams based on the count or count-of-count statistics. In the framework of the continuous space language model (CSLM), the $n$-gram probability is estimated based on the word representations in a continuous space embedding. Words which are somehow related are expected to reside in nearby neighborhoods in the continuous space representation. Thus interpolated probabilities for rare and unseen events can be more effectively computed. As compared to the traditional methods, the CSLM has been shown to provide better generalization capabilities. The idea of CSLM can be realized by means of neural networks [4]−[18], where the projection of words into a continuous space and the estimation of their probabilities can be conducted by feedforwarding the input word indices across the network.

Here, we refer to the original implementations of feedforward neural network language models [4], [7] as CSLM. In this type of implementations, the input to the neural network is a series of vectors with 1-of-$N$ binary encoding, where $N$ is the vocabulary size, and each vector indexes a word in the history-context. At the first hidden layer, each binary vector is projected into a $P$-dimensional continuous space (where $P \ll N$). The projection matrix, i.e. the weights of the connections between the input layers and the first hidden layer, are common to all the history-words. After projection, the obtained vectors are concatenated in sequence to provide a continuous space representation of the history-context. Second hidden layer (of size $H$) and the output layer (of size $N$) are used to estimate the corresponding $n$-gram probabilities based on the history-context representation. The output layer gives the posterior probabilities for each target-word in the vocabulary. Further details can be found in [4], [7].

4. **Continuous space implementation of the TDTO language model**

We follow a similar implementation strategy to the one used in the CSLM [4], [7] to propose a continuous space version of the TDTO model. The motivation is to learn better TDTO model representation based on TD and TO continuous space representations. We refer to this model as the continuous space TDTO (cTDTO) language model. The proposed neural network architecture is shown in Figure 1.

![Figure 1: Neural network architecture for the implementation of the cTDTO language model.](image)

The inputs to the neural network are two vectors, which encode the TD and the TO model components, respectively. Each coding vector has a dimension of $N$ (vocabulary size), and the $j$-th element in the vector corresponds to the $j$-th word in the vocabulary. The TD coding vector describes the positions of words in the history-context according to the following encoding scheme:

$$x_{\beta}^{\beta}(w_{i-k}) = \begin{cases} 1 & \text{if } w_{i-k} = v_j \\ 0 & \text{otherwise} \end{cases}$$ (3)
where \( z_j^O \) denotes the \( j \)-th element in the TD encoding vector \( \mathbf{z}^D \), \( v_j \) denotes the \( j \)-th vocabulary word, and \( w_{i-k} \) denotes the history-word at distance \( k \) from the target word. The parameter \( a \) controls the decay rate of the position influence as the history-word goes far, i.e. \( a \geq 1 \). Note that for words located beyond the considered history-context, we assume \( k = \infty \) thus \( z_j^O = 0 \).

The TO encoding vector, on the other hand, describes the occurrences of words in the history-context according to the following encoding scheme:

\[
x_j^O(w_{i-k}) = \begin{cases} \#(w_{i-k} \in h) & \text{if } w_{i-k} = v_j \\ 0 & \text{otherwise} \end{cases}
\]

(4)

where \( x_j^O \) denotes the \( j \)-th element in the TO coding vector \( \mathbf{z}^O \), \( v_j \) denotes the \( j \)-th vocabulary word, and \( \#(w_{i-k} \in h) \) denotes the count of word \( w_{i-k} \) in the considered history-context \( h \).

At the first hidden layer, the TD and TO input vectors, \( \mathbf{z}^D \) and \( \mathbf{z}^O \), are projected into their respective continuous space representations:

\[
\mathbf{c}^D = \mathbf{Q}^D \mathbf{z}^D
\]

(5)

\[
\mathbf{c}^O = \mathbf{Q}^O \mathbf{z}^O
\]

(6)

where \( \mathbf{Q}^D \) and \( \mathbf{Q}^O \) are the projection matrices of the TD and TO vectors, while \( \mathbf{c}^D \) and \( \mathbf{c}^O \) are the resulting continuous space representations, respectively. The dimensions of both projection matrices are \( P \times N \).

The representation vectors are then concatenated to be used as input to the second hidden layer. The second hidden layer uses hyperbolic tangent activation function as non-linearity for the estimation of the TDTO probability:

\[
\mathbf{d} = \tanh(\mathbf{M} \mathbf{c} + \mathbf{b})
\]

(7)

where \( \mathbf{c} \) denotes the concatenation of the TD and TO representation vectors:

\[
\mathbf{c} = [\mathbf{c}^D \mathbf{c}^O]' \tag{8}
\]

\( \mathbf{M} \) is the weight matrix and \( \mathbf{b} \) is the bias term vector between the first and second hidden layers. The dimensions of the weight matrix and the bias vector are \( H \times P \) and \( H \times 1 \).

The output layer provides the probability distribution of the target-word:

\[
\mathbf{o} = \mathbf{V} \mathbf{d} + \mathbf{r}
\]

(9)

where \( \mathbf{V} \) and \( \mathbf{r} \) are the weight matrix and the bias vector between second hidden layer and the output layer. In the output layer, the softmax function is used to normalize the probabilities to sum to one:

\[
\mathbf{p} = \frac{\exp(\mathbf{o})}{\sum_{l=1}^N \exp(o_l)} \tag{10}
\]

The \( \mathbf{p} \) vector represents the probabilities of all the words in the vocabulary given the TD and TO input vectors. \( ||\mathbf{p}|| = N \).

In the proposed feedforward neural network implementation, the computational complexity associated to a word probability is determined by the size of the hidden layers and the output layer, shown as follows:

\[
2 \times P \times H + H \times H \times N + N
\]

(11)

Here, the complexity is defined as the number of multiply-and-add operations. Note that this complexity estimate does not include the projection of the TD and TO encoding vectors into the continuous space as it involves table look-up and, therefore, the computation is trivial. As compared to the complexity of the CSLM reported in [7], which grows with regard to the history length, the cTDTO model complexity is constant regardless the size of the history length.

For the training, the backpropagation algorithm is used to learn the neural network parameters iteratively. During the training, the TD and TO input vectors are presented to the network and the weights are adjusted to minimize the cross-entropy between the actual generated output and the desired reference output.

The proposed cTDTO model has several practical advantages, as compared to the TDTO model [1]–[3]. First, probability estimates can be computed faster by feedforwarding the inputs across the network, in contrast to the TDTO model which involves the calculation of the normalization term \( Z(h) \), as shown in (2), which is very expensive from the computational point of view. Furthermore, the TDTO model requires some smoothing parameters to be assigned intuitively, which are difficult to optimize in a systematic way. Finally, the cTDTO model needs no development phase once it is trained, as opposed to the TDTO model which requires the component weights to be further optimized.

5. Experimental results

To assess the improvement contributed by the neural network implementation, we compared the perplexity of the cTDTO model to the baseline TDTO model. Also, the performances of both models in a speech recognition re-ranking task were compared.

Model perplexities were evaluated on the BLLIP Wall Street Journal (WSJ) corpus [30]. The training set contains one tenth of the 87’s subset, which accounts for about 1.7M words. Rather than using the entire 87’s subset for the training, as in our previous works [1]–[3], a smaller training set was used to speed up the training of the neural network. Both the development and test sets, each containing 32K words, were selected from the 88’s subset. The vocabulary consists of a set of 5K words, which are adopted from the Aurora-4 task [31].

The TDTO model was built following [1]–[3], the training set was used to train the n-gram, TD, and TO model components, while the development set was used to optimize the weights: \( \beta_H, \beta_D \), and \( \beta_T \). For the cTDTO model, the feedforward neural network consisted of 50 nodes in each TD and TO projection layers, i.e. \( P = 50 \), and 500 nodes in second hidden layer, i.e. \( H = 500 \). The sizes of the input and output layers follow the size of the vocabulary (5K), i.e. \( \mathbf{N} \). The training set was used for conducting the training while the development set was used for validation during the training. The feedforward neural network used in the cTDTO model was modified from the CSLM toolkit [32].
5.1. Perplexity evaluation

We compared the perplexities of the cTDTO model with the baseline TDTO model to examine the benefit of the neural network implementation. We considered history lengths from one to ten words. The results are shown in Figure 2.

![Figure 2: Perplexities of the cTDTO model as compared to the TDTO model. The cTDTO model showed lower perplexities in all considered history lengths, from two to ten. The relative reduction (Rel. Red.) is referred by the right vertical axis.](image)

Note that the horizontal axis refers to the length of the history from which the TD and the TO are captured. For history length of one, the comparatively low perplexity of the TDTO model was in fact contributed by the n-gram component in the model, see (2), which in this case is a trigram model.

As seen in Figure 1, the cTDTO model consistently showed lower perplexities, as compared to the TDTO model, in all considered history lengths (except for the case of history length one). More specifically, the cTDTO model showed higher gain in longer history, i.e. 2.4% of perplexity reduction (from 89.8 to 87.7) at the history length of two, and monotonically increased to 9.2% (from 85.6 to 77.7) at the history length of ten. Greater relative reduction of the perplexity as according to the history length implies the data scarcity problem in modeling longer history-context was more effectively tackled by the cTDTO model. The relative reduction is referred by the right vertical axis in Figure 2. Overall, the neural network based implementation has been shown to provide a better generalization capability with respect to the original TDTO model implementation.

5.2. Speech recognition

We also compared the performance of the cTDTO model with the TDTO model in a speech recognition N-best re-ranking task. The experiment was conducted on the clean subset of the Aurora-4 corpus [31]. The speech recognition system, including the acoustic model and the language model, was configured in the same way as in our previous work [2] [3]. For each utterance, a 200-best list of hypotheses was generated by using the HTK's HVite [33] decoder and then, we recomputed the language scores by using the TDTO and the cTDTO models. Hypotheses with the best score were selected as the recognized utterances. The WER obtained with the TDTO model served as the baseline to be compared with the cTDTO model result. The cTDTO model and the TDTO model were trained from the BLLIP WSJ corpus, as in the previous experiment. The resulting word error rates (WER) are shown in Figure 3.

![Figure 3: WER results of the cTDTO model as compared to the TDTO model. The cTDTO model showed lower WER in all considered history lengths, from two to ten. The relative reduction (Rel. Red.) is referred by the right vertical axis.](image)

The WER results are consistent with the perplexity evaluation results shown in Figure 2. For all the considered history lengths, except for history length of one, the cTDTO model outperformed the TDTO model. Particularly at the history length of seven, the WER has been reduced up to 12.9% relatively (from 9.8% to 8.6%). The relative reduction of the WER is referred by the right vertical axis. The results confirm the effectiveness of the continuous space implementation in the cTDTO modeling.

6. Conclusions and future work

We have presented and evaluated a continuous space implementation of the TDTO (term-distance term-occurrence) language model. The presented implementation, which is referred to as cTDTO, is based on a feedforward neural network architecture. Due to the superior smoothing capabilities offered by the neural network, the cTDTO model outperformed the TDTO baseline model in both perplexity evaluation and speech recognition re-ranking task. Evaluated on the WSJ corpus, the cTDTO model yielded up to 9.2% lower perplexity as compared to the TDTO model. In the Aurora-4 corpus, the cTDTO model obtained up to 12.9% relatively lower WER.

As the future work, we will examine a deep neural network [12], [34] implementation of the TDTO model as the extension to the currently applied feedforward neural networks which relies on only a single hidden layer to learn the non-linearity of the probability function. The deep neural network would learn a higher level representation of the TD and TO which may lead to a better modeling of the TDTO model.

Also, we will look into the problem of speeding up the training process by limiting the size of the output layer to only the shortlisted words [7] and hierarchical partitioning nodes at the output layer [5], [11]. These techniques are currently used in various neural network language modeling to enhance the training time of the neural networks. Also, we will study the possibility of resampling the training data [7], [9], [10], [35] in order to include larger data set for training the cTDTO model.

7. Acknowledgement

This work is supported by DSO funded project MAISON DSOCL14045.
8. References


