Exploiting long context using joint distance and occurrence information for language modeling

Chong, Tze Yuang

2018

Chong, T. Y. (2018). Exploiting long context using joint distance and occurrence information for language modeling. Doctoral thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/75876>

<https://doi.org/10.32657/10356/75876>

EXPLOITING LONG CONTEXT USING JOINT
DISTANCE AND OCCURRENCE INFORMATION
FOR LANGUAGE MODELING

CHONG TZE YUANG

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

2018

EXPLOITING LONG CONTEXT USING JOINT
DISTANCE AND OCCURRENCE INFORMATION
FOR LANGUAGE MODELING

CHONG TZE YUANG

School of Computer Science and Engineering

A thesis submitted to Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2018

Acknowledgments

I would like to express my sincere thanks and appreciation to Dr. Chng Eng Siong, Prof. Li Haizhou, and Dr. Rafael E. Banchs for their supervision and guidance. I have benefited greatly from their invaluable advice. Their assistance allows me to pursue my interests and explore new research areas. I am grateful for their constant encouragement and their perspectives on life at large.

Contents

Acknowledgments	i
List of Figures	vii
List of Tables	ix
List of Abbreviations	xiii
Abstract	xv
1 Introduction	1
1.1 What is a language model?	1
1.2 Motivations	2
1.3 Objectives	3
1.4 Contributions	4
1.5 Thesis structure	5
2 An Overview of Language Modeling	7
2.1 Background	8
2.1.1 Definition of Language Models	8
2.1.2 Applications of Language Models	8
2.1.3 Evaluations of Language Models	10
2.2 Types of Language Models	11

2.2.1	<i>N</i> -gram Model	13
2.2.2	Distant-Bigram Model	18
2.2.3	Skip-Gram Model	19
2.2.4	Trigger Model	20
2.2.5	Bag-of-Words Model	22
2.3	Model Combination	24
2.3.1	Linear Interpolation	25
2.3.2	Log-Linear Interpolation	26
2.3.3	Maximum Entropy	26
2.4	The Inconsistency Problem	27
2.5	Summary	28
3	Term-Distance Term-Occurrence Language Model	31
3.1	Background	32
3.1.1	Model Combination	32
3.1.2	Parameters Inconsistency	33
3.2	Term-Distance and Term-Occurrence	34
3.2.1	Joint Modeling of <i>N</i> -gram and BOW	35
3.2.2	Modeling Distance and Occurrence of Word-Pairs	36
3.2.3	Model Generalization	39
3.2.4	Model Component Computation	41
3.3	Summary	45
4	Evaluation and Analysis of the TDTO model	47
4.1	Perplexity Evaluation	48
4.1.1	Corpora	48
4.1.2	Model Settings	49
4.1.3	Experiment 1: TDTO for modeling long context	51
4.1.4	Experiment 2: TDTO vs. LLI of <i>n</i> -gram and BOW	55

4.1.5	Experiment 3: TDTO vs. Distant-Bigram and Trigger	57
4.2	Applications of the TDTO Model	62
4.2.1	Application 1: Speech Recognition	62
4.2.2	Application 2: Text Classification	64
4.2.3	Application 3: Word prediction	67
4.3	Summary	69
5	The Neural Network Approach to TDTO Language Modeling	71
5.1	Background	72
5.2	Neural Network based TDTO Model	74
5.2.1	Motivations	75
5.2.2	Architecture of the NN-TDTO model	76
5.2.3	Complexity of the NN-TDTO model	79
5.3	Experiments	80
5.3.1	Perplexity Evaluation	81
5.3.2	Speech Recognition	82
5.3.3	NN-TDTO vs. NN-NG	84
5.4	Summary	86
6	Conclusions and Future Work	89
6.1	Conclusions	89
6.2	Future Work	91
6.2.1	Class-based TD	91
6.2.2	Convolutional Neural Network based TDTO Model	92

List of Figures

2.1	Types of language models and their respective smoothing paradigms . . .	13
2.2	A fourgram model represents the history as the immediate three-word sequence.	14
2.3	A distance-4 bigram model represents the history by the word located in the 4-th position.	19
2.4	A skip-gram model represents the history as a sub-sequence of words in the history.	20
2.5	A trigger model represents the history as the binary occurrence of the correlated word.	21
2.6	A BOW model represents the history as an unordered list of words. . . .	22
2.7	A fourgram model is combined with a BOW model to exploit longer context in the history to estimate probability of the word “ <i>market</i> ”.	25
3.1	Inter-dependencies between n -gram and BOW are not captured by LI and LLI.	34
4.1	On the WSJ and SWB corpora, the perplexities of the TDTO model were reduced up to 11.2% and 6.5%, respectively, when longer context was considered. Also, the reduction of perplexities within the trigram context indicates that the TD and TO components are capable of capturing complementary information from the trigram context.	51

4.2	Optimal weights for the trigram (3G), TD, and TO model components as corresponding to the perplexities depicted in Figure 4.1, for the WSJ and SWB corpora.	52
4.3	The perplexities of the n -gram models (NG) of orders from two to six, i.e. context lengths from one to five, as compared with the TDTO models that share the same context. The results imply that the TDTO model component is capable of capturing complementary information in the n -gram context that the n -gram model alone is not able to capture.	55
4.4	Resulting perplexities for the TDTO and LSI models of context lengths from zero to ten. On both corpora, the TDTO model showed lower perplexities as compared to the LSI model.	56
4.5	Resulting perplexities for TDTO and DBG models of history lengths from zero to ten. On the WSJ corpus, both results are comparable. On the SWB corpus, the TDTO model reduced the perplexity up to 6.5% while the DBG model showed only marginal improvement.	60
4.6	On the WSJ corpus, perplexity reductions to the n -gram models (NG), when interpolated with TDTO, DBG and TGR models looking at the same history-context lengths. On the SWB, TDTO model outperformed both DBG and TGR models.	61
4.7	Word error rate (WER) resulting from n -gram models (NG) of order from two to six, and the respective interpolated TDTO models. The TDTO models consistently improved the WER obtained by using the n -gram alone, with relative improvements (RI) of 4.6-9.7%.	64
5.1	Neural network architecture for the implementation of the NN-TDTO language model.	76

5.2	The perplexities of the NN-TDIO and TDIO models. As compared to the baseline TDIO model, the NN-TDIO model consistently showed lower perplexities for all considered history lengths. Note that for the context length of one, the comparatively low perplexity of the TDIO model was contributed by the trigram component that considers a history length of 2. In this case, it is not a fair comparison. The relative reduction (Rel. Red.) percentage is depicted in the right vertical axis.	82
5.3	WER results of the NN-TDIO model as compared to the TDIO model. The NN-TDIO model showed lower WER in all considered history lengths, from two to ten. The relative reduction (Rel. Red.) is depicted in the right vertical axis.	83
5.4	The perplexities of the NN-TDIO and NN-NG models for history lengths from one to ten. The performances of both models are not significantly different.	85
5.5	The perplexities of the NN-TDIO and NN-NG models with the projection layers's dimensions increased to 100 (top) and 150 (bottom). Although both models did not lower perplexities for history length exceeds seven, the NN-TDIO model is more resistant to the noise incurred by the long context.	86
5.6	the average per-epoch training time results of the NN-TDIO and NN-NG models. The training time of the NN-TDIO model did not change significantly with respect to the history length, as opposed to the NN-NG model's training time that increased linearly.	87

List of Tables

2.1	Hypotheses produced by a speech recognition system contain homophones, i.e. “ <i>chip</i> ” and “ <i>cheap</i> ”.	9
2.2	Types of language models and their respective representations, i.e. $\hat{h}$, of the history	13
4.1	Partitioning of the WSJ corpus	49
4.2	Partitioning of the SWB corpus	49
4.3	F1 score for each category in the Reuter-21578 dataset as obtained by the unigram, sixgram, 1G-TD, and 1G-TO models.	66
4.4	F1 score in the polarity dataset as obtained by the unigram, sixgram, 1G-TD, and 1G-TO models.	66

List of Abbreviations

BLEU	bilingual evaluation understudy
BOW	bag-of-words
CNN	convolutional neural network
FNN	feedforward neural network
IDF	inverse document frequency
IR	information retrieval
KL	Kullback-Leibler
LI	linear interpolation
LDA	latent dirichlet allocation
LLI	log-linear interpolation
LSI	latent semantic indexing
LSTM	long-short term memory
MI	mutual information
MLE	maximum likelihood estimation
NLP	natural language processing
NN	neural network
pLSI	probabilistic latent semantic indexing
PPL	perplexity
POS	part-of-speech
RNN	recurrent neural network

SWB	Switchboard
TD	term-distance
TDTO	term-distance term-occurrence
TF	term frequency
TO	term-occurrence
WER	word error rate
WSJ	Wall Street Journal

Abstract

This thesis investigates an approach to exploiting the long context based on the information about the distance and occurrence. By modeling the joint event of distance and occurrence, this approach attempts to incorporate the inter-dependencies into the model, such that information captured from the long context can be more optimally made use. This thesis addresses the problem with the conventional language modeling approaches that tend to neglect the inter-dependencies.

Based on the proposed approach, a novel language model, referred to as the term-distance term-occurrence (TDTO) model, is formulated. The TDTO model estimates probabilities based on the events of term-distance (TD) and term-occurrence (TO) that correspond to the distances and occurrences of words in the context. By expressing the TDTO model in terms of a log-linear interpolation framework, the impact of the TD and TO towards the final estimation can be tuned. Specifically, as the TD events, i.e. positions, within a long context are likely rare or unseen, the weight of the TD component can be tuned down accordingly to alleviate the data scarcity problem.

Through a series of experiments, the TDTO model has been shown to be capable of exploiting the long context to reduce the perplexities of the language models. On the BLLIP Wall Street Journal (WSJ) and Switchboard-1 (SWB) corpora, perplexity reductions up to 11.2% and 6.5% were obtained, with the context lengths of seven and eight, respectively. In addition, the TDTO model has been shown to outperform other conventional models used to exploit the long context, such as the distant-bigram, trigger and BOW models – the TDTO model consistently showed lower perplexities. The ap-

plicability of the TDTO model has been examined on several tasks, such as the speech recognition, text classification and word prediction. The TDTO model has been shown to improve the baseline performance on all the considered tasks.

Furthermore, this thesis proposes a neural network implementation of the TDTO model. The aim is to provide a better smoothing mechanism for TDTO modeling. The resulted model, referred to as the neural network based TDTO (NN-TDTO) model, has been empirically shown to outperform the baseline TDTO model in both perplexity and speech recognition accuracy. On the WSJ corpus, the NN-TDTO model yielded up to 9.2% lower perplexity as compared to the TDTO model. On the Aurora-4 speech recognition task, the NN-TDTO model obtained up to 12.9% relatively lower word error rate.

Chapter 1

Introduction

1.1 What is a language model?

Language is a system of communication consisting of the use of words in a structured and conventional way¹. Words are organized according to a set of syntactic rules to form sequences. By putting words of different semantic relations together in a context, a thought is conveyed. In addition, pragmatic and discourse govern how words are chosen and combined so that the context is sensible to the situation. Language models are built to capture such regularities of words usage. By encapsulating the variation of words in different arrangements, a language model assigns probability to a given sequence of words indicating how likely it would be seen in the language.

Language models provide useful linguistic knowledge to facilitate various applications, such as speech recognition, machine translation and information retrieval. The speech recognition systems, for example, rely on the language models to constrain the search to the linguistically well-formed transcripts. When distinguishing acoustically ambiguous words, the language models select words that are more probable to happen based on the syntactic and the semantic in the context.

¹Oxford dictionaries: <http://www.oxforddictionaries.com/>

1.2 Motivations

The probability of a word sequence can be computed as a product of probabilities of the words in the sequence, where each word is conditioned on the preceding context, known as the history. However, exploiting the entire history is impractical. A history is typically reduced to a representation that consists of only simple events, such as sub-sequence of words or bag-of-words. The n -gram models [Bahl et al., 1983, Nádas, 1984], for example, exploit the sequence of $n - 1$ words in the immediate context. As n is usually limited low, e.g. $n \leq 4$, the n -gram models estimate probabilities based on only short context – much information about the history is neglected. Exploiting longer context in the history can be performed by combining an n -gram model with another type of language model, e.g. the distant-bigram model [Huang et al., 1993, Simons et al., 1997, Klakow, 1998], the skip-gram model [Chelba and Jelinek, 2000, Siu and Ostendorf, 2000, Guthrie et al., 2006], the trigger model [Lau et al., 1993b, Rosenfeld, 1996] or the bag-of-words (BOW) model [Bellegarda, 1998, Coccaro, 2005, Novak and Mammone, 2001]. These models consider representations that span longer context but incur less complexity in the models.

Combining language models provides a means to incorporate multiple representations of the history to estimate probabilities. However, since each representation is modeled in separate, inter-dependencies among the considered events would be neglected. To be more specific, models are combined by using a simple averaging technique, e.g. linear interpolation (LI) [Jelinek and Mercer, 1980, Brown et al., 1992] or log-linear interpolation (LLI) [Klakow, 1998], which takes no measure to exploit the inter-dependencies. As a result, parameters in the combined model are not consistent to the statistics in the training data – there will be events over-counted or under-counted. In the case of combining an n -gram model with another model to exploit the long context, e.g. a BOW model, information captured from the history will be sub-optimally used. There are more complex techniques available for model combination, such as history-dependent interpolation [Bahl et al., 1983, Chen and Goodman, 1996] and maximum entropy (ME)

[Simons et al., 1997, Lau et al., 1993b, Rosenfeld, 1996], where by fine-tuning the model parameters, inter-dependencies can be recovered. However these methods are computational intensive.

1.3 Objectives

The objective of this thesis is to formulate an approach to exploiting the long context based on the joint event of multiple representations of the history. The aim is to incorporate the inter-dependencies into the model, as a means to better utilize information captured from the context. This thesis addresses one of the weaknesses in the approaches used to combine language models, i.e. LI and LLI, where the inter-dependencies among the events tend to be neglected.

Estimating a joint model is non-trivial due to the enormous number of parameters incurred by the permutations of the events. Conventional methods used for smoothing are applicable to models that exploit only a single type of event, e.g. n -gram. Modeling the joint of multiple types of events demands a novel strategy for parameter estimation. This thesis considers decomposing the probability function, such that the complexity in the model is segregated into components of lesser complexities. Let $\theta_1, \theta_2, \theta_3, \theta_4, \dots$ be the events, defined in the sample space of word sequences, used to predict a given word w . If the complexity of modeling each predictor decreases monotonically, such that

$$O(P(w|\theta_1)) > O(P(w|\theta_2)) > O(P(w|\theta_3)) > O(P(w|\theta_4)) > \dots \quad (1.1)$$

the joint probability function can be decomposed, based on the chain rule, as follows:

$$P(w|\theta_1, \theta_2, \theta_3, \theta_4, \dots) \propto P(w)P(\theta_1|\theta_2, \theta_3, \theta_4, \dots, w)P(\theta_2|\theta_3, \theta_4, \dots, w)P(\theta_3|\theta_4, \dots, w)\dots \quad (1.2)$$

Predicting w given the joint of $\theta_1, \theta_2, \theta_3, \theta_4, \dots$ can be computed as a product that comprises components of less specific events, i.e. $P(\theta_2|\theta_3, \theta_4, \dots, w)$, $P(\theta_3|\theta_4, \dots, w)$ and etc. Under the situation when an instance of $\theta_1, \theta_2, \theta_3, \theta_4, \dots$ is rare or unseen, the probability can still be estimated more reliably by using the components of less specific events.

This thesis considers an approach to modeling the joint of two types of events: (1) position and (2) occurrence, where the former refers to the discrete locations of words, while the latter refers to the binary occurrences of words. Note that these two types of events are inter-dependent – both describe the presence of words within a context. Moreover, as the position event is a more specific descriptor as compared to the corresponding occurrence event, they serve the suitable choices of linguistic events to explore the problem. Based on the decomposition formulation mentioned in Equation (1.2), predicting words given a joint of position and occurrence events can hence be computed as follows:

$$P(\text{word}|\text{position}, \text{occurrence}) \propto P(\text{word})P(\text{position}|\text{occurrence}, \text{word})P(\text{occurrence}|\text{word}) \quad (1.3)$$

This thesis aims to realize an approach to modeling the joint event of position and occurrence to predict words – by decomposing the model into two components, one exploits the position while the other one exploits the occurrence. Nevertheless, the main challenge for this study is to model the position-given-occurrence component which is novel in the field of language modeling, particularly to devise a smoothing mechanism to handle the rare and unseen events.

1.4 Contributions

Throughout the study of this thesis, a novel approach to exploiting the joint event of position and occurrence has been proposed. A language model, referred to as the term-distance term-occurrence (TDTO) model has been formulated. Following the formulation in Equation (1.3), the TDTO model comprises two main model components: term-distance (TD) and term occurrence (TO), which are used to exploit the positions and occurrences of words, respectively. By using the TD and TO model components to exploit the long context, experiment results have shown that information about the long context is captured – the proposed TDTO model has been shown to improve the trigram

models both in terms of perplexity and speech recognition accuracy. As compared to other types of language models, such as high-order n -gram, distant-bigram, trigger and BOW models, the TDTO model has shown to perform better. These findings have been shared in numerous publications:

Chong, T.Y., Banchs, R.E., Chng, E.S. & Li, H., 2013, Modeling of term-distance and term-occurrence information for improving n -gram language model performance, in Proc. ACL, pp. 233–237.

Chong, T.Y., Banchs, R.E., Chng, E.S. & Li, H., 2014, Improving language modeling by using distance and co-occurrence information of word-pairs and its application to LVCSR, in Proc. ICASSP, pp. 4883–4887.

Chong, T.Y., Banchs, R.E., Chng, E.S. & Li, H., 2015, Decoupling word-pair distance and co-occurrence information for effective long history context language modeling, Trans. TASLP, 23(7): 1221-1232.

Furthermore, the smoothing problem of the TD and TO model components has been addressed by using a feed-forward neural network. The resulted model, referred to as the neural network based TDTO (NN-TDTO) model, has shown to improve the perplexity of the baseline TDTO model and when being evaluated on a speech recognition task, lower WER was obtained. This finding has been published in the following paper

Chong, T.Y., Banchs, R.E., Chng, E.S. & Li, H., 2015, TDTO language modeling with feedforward neural networks, in Proc. INTERSPEECH, pp. 233–237.

1.5 Thesis structure

The rest of this thesis is organized as follows:

Chapter 2 provides a review on language modeling approaches. It covers a literature survey of different techniques used to build and combine language models.

Chapter 3 proposes the TDTO model. It discusses the motivation as well as the formulation of the TDTO modeling. The evaluation of the TDTO model is presented in Chapter 4.

Chapter 5 proposes the NN-TDTO model. It addresses the smoothing problem with the original TDTO model. The evaluation of the NN-TDTO is presented.

Chapter 6 concludes this thesis and discusses the future works.

Chapter 2

An Overview of Language Modeling

The role of a language model is to estimate probabilities of word sequences. By using the chain rule, a language model can be realized as a conditional probability function that computes probabilities of words based on the preceding context, i.e. the history. In this way, by considering a simple representation of the history, a language model can be feasibly built. Generally, language models can be classified into: n -gram, distant-bigram, skip-gram, trigger and bag-of-words, each class considers a specific representation of the history. In order to provide more accurate probabilities, language models of different types are combined as a means to capture richer information about the history.

In this chapter, language modeling approaches are reviewed. Section 2.1 provides an introduction to the language models: the definition, the evaluation and the applications. Section 2.2 discusses the techniques used to build each type of the language models, i.e. n -gram, distant-bigram, skip-gram, trigger and bag-of-words. Section 2.3 reviews three techniques used to combine the language models: linear interpolation, log-linear interpolation and maximum entropy. In the last section, one of the weaknesses of these combination techniques is highlighted – the inter-dependencies among the events that are considered by the component models tend to be neglected. As a result, the combined model performs sub-optimally.

This problem will be addressed in the next chapter.

2.1 Background

2.1.1 Definition of Language Models

A language model is a mathematical representation of the natural language phenomenon. It acquires and uses knowledge of how words are organized in sequence based on syntactic rules and semantic relations between words. Linguistic knowledge is encapsulated by the language models as a probability distribution over word sequences, where probability would be assigned to a given word sequence indicating how likely it would be seen in the language.

Typically, words are assumed to occur based on only history-words, thus the probability of a given word sequence, say $s = (w_1, w_2, \dots, w_m) = w_1^m$, can be computed by using the chain rule, as follows.

$$P(s = w_1^m) = \prod_{i=1}^m P(t = w_i | h = w_1^{i-1}) \quad (2.1)$$

where t denotes an arbitrary target-word w_i in sequence s , and h denotes the corresponding history $w_1^{i-1} = (w_1, w_2, \dots, w_{i-1})$.

As shown in Equation (2.1), language models can be defined more specifically as a product of conditional probabilities, each predicts a word in the sequence given the history. Such definition allows probabilities to be estimated more feasibly by approximating the history as a reduced representation, such as the sub-sequence of words or the bag-of-words. Further discussion will be presented in Section 2.2.

2.1.2 Applications of Language Models

Language models provide linguistic knowledge to serve many applications. In the interaction between computers and language, natural language processing (NLP) modules rely on the language models to analyze, understand and derive useful information from word sequences. Particularly, the probability provided by the language models indicates

how linguistically well-formed a given word sequence is, to assist decision making in the applications.

In the speech recognition systems [Katz, 1987, Jelinek, 1990], for example, language models are used to constrain the search space such that only word sequences that are more likely to occur would be considered as solutions. Moreover, language models resolve the pronunciation ambiguities among word candidates by taking information about the context, e.g. the history, into account. Consider the two hypotheses of word sequences, as shown in Table 2.1, produced by a speech recognition system given an input utterance, which each contains a homophone, i.e. “*chip*” and “*cheap*” respectively. As it is more probable to observe “*chip*” than “*cheap*” given the context, the first hypothesis will be assigned a higher probability.

Table 2.1: Hypotheses produced by a speech recognition system contain homophones, i.e. “*chip*” and “*cheap*”.

Hypothesis	Word sequence
#1	<i>a collapse in blue <u>chip</u> stocks could be a boon for the rest of the market</i>
#2	<i>a collapse in blue <u>cheap</u> stocks could be a boon for the rest of the market</i>

For applications such as machine translation [Brown et al., 1990] and handwriting recognition [Marti and Bunke, 2001], language models are applied in the same fashion as in the speech recognition systems to govern the search for the optimal solutions. However, for these two types of applications, the language models are used to disambiguate word candidates of similar semantic and orthography.

On the other hand, for applications of information retrieval [Ponte and Croft, 1998, Song and Croft, 1999], syntactic parsing [Manning and Carpenter, 1997, Charniak, 2001], and text classification [Collins, 2003], language models are used to indicate how likely a given word sequence belongs to a specific class of language, such that by exploiting the discrepancies among probability values of different classes, the given word sequence or word would be assigned a label or a score of relevancy.

2.1.3 Evaluations of Language Models

A language model is evaluated to examine how accurate the estimated distribution is with respect to the true distribution, which can be measured by computing the cross entropy between the two distributions over all word sequences, shown as follows.

$$H(P, P_{\text{true}}) = - \sum_{s \in S} P_{\text{true}}(s) \log P(s) \quad (2.2)$$

where P and P_{true} denote the estimated and the true distributions, respectively.

By definition, the cross entropy measures how uncertain a word sequence s is to be predicted by P . The metric depends on the complexity in the language and the statistical distance between P and P_{true} .

$$H(P, P_{\text{true}}) = H(P_{\text{true}}) + D_{\text{KL}}(P_{\text{true}} || P) \quad (2.3)$$

where $H(P_{\text{true}})$ denotes the self entropy of the distribution P_{true} , i.e. the inherited complexity in the language; while $D_{\text{KL}}(P_{\text{true}} || P)$ denotes the Kullback-Leibler (KL) distance between the distributions P_{true} and P . As the complexity in a language is typically fixed, the quality of a language model relies on how close the estimated distribution is to the true distribution.

By incorporating the chain rule assumption as shown in Equation (2.1), the cross entropy can be approximated by

$$H(P, P_{\text{true}}) \approx - \frac{1}{K} \sum_{i=1}^K \log P(t = w_i | h = w_1^{i-1}) \quad (2.4)$$

where $(w_1, \dots, w_i, \dots, w_K)$ denotes a set of data of which P_{true} can be observed, referred to as the test-set.

For many applications, cross entropy is expressed in terms of perplexity (PPL) which can be interpreted as the branching factor – a model of lower PPL is said to be more certain in predicting next words [Bahl et al., 1977, Brown et al., 1992]. The PPL of a language model can be computed as follows.

$$\text{PPL} = 2^{-\frac{1}{K} \sum_{i=1}^K \log_2 P(t=w_i | h=w_1^{i-1})} \quad (2.5)$$

A language model can also be evaluated based on the performance of the NLP application which the model is applied to. In such cases, language models are evaluated by metrics varying across different applications, for example word error rate (WER) for speech recognition, bilingual evaluation understudy (BLEU) score for machine translation, F-score for information retrieval, etc.

2.2 Types of Language Models

The role of a language model is to estimate probabilities of word sequences. One issue with building language models is the data scarcity problem – because the number of possible word sequences is practically infinite, any large amount of data is inadequate to observe an accurate distribution of uncommon events. Modeling word sequences of length m would require $|V|^m$ number of parameters, where $|V|$ denotes the vocabulary size. Even for a medium size vocabulary, e.g. $|V| = 50\text{k}$, modeling sequences of five words, i.e. $m = 5$, would incur an amount of parameters in the order of 10^{23} . Acquiring adequate data to adapt such enormous amount of parameters is impractical.

Typically, the probability of a word sequence, say $s = (w_1, w_2, \dots, w_m)$, is computed by using the chain rule:

$$P(s = w_1^m) = \prod_{i=1}^m P(t = w_i | h = w_1^{i-1}) \quad (2.6)$$

where t denotes an arbitrary target-word w_i in sequence s , and h denotes the corresponding history w_1^{i-1} . The benefit of estimating probabilities based on the events of word-history, i.e. $(h = w_1^{i-1}, t = w_i)$, is that the data scarcity problem can be tackled by considering an equivalent class of the history:

$$\hat{h} = \Phi(w_1^{i-1}) \quad (2.7)$$

where $\Phi(\cdot)$ denotes a function that gives a less sparse approximation of the full-history, i.e. $\hat{h}$, such as a sub-sequence of words or a bag-of-words. As $\hat{h}$ is typically a reduced

representation of h where its statistics can be more easily exploited, building language models becomes more feasible.

In general, a representation of the history can be obtained by simplifying the word sequence in two ways:

1. **Truncating the word sequence:** Words that are less contributive, as determined under certain assumptions, are dropped from the word sequence giving a shorter sub-sequence. The n -gram models, for example, exploit $(n - 1)$ words in the immediate context hence limit the number of model parameters to $|V|^n$. Other types of language models, such as the distant-bigram model and the skip-gram model, share the same spirit by considering a shorter sequence, i.e. $n < i$, in the history hence reducing the number of model parameters in orders of magnitude, i.e. $|V|^n \ll |V|^i$.
2. **Disregarding the order of words:** One may also ignore the order of the words in the history to approximate the word sequence w_1^{i-1} as the bag-of-words (BOW) or the trigger words. These two types of representations consider only the binary occurrences of words, which allow for the number of model parameters to be drastically reduced. For example, modeling the BOW of the contexts of n -word requires only $|V|C_n^1$ number of parameters, i.e. $|V|C_n \ll |V|^i$.

Based on the type of representation considered, i.e. how the word sequence is simplified, language models can be classified into categories of: n -gram, distant-bigram, skip-gram, trigger, and bag-of-words, as shown in Table 2.2.

For each type of language model, smoothing is applied to improve generalization, particularly, to assign more accurate probabilities to the rare and unseen events. Figure 2.1 depicts an overview of the smoothing methods adopted by each model type, which will be discussed in the following sub-sections.

¹In fact, a BOW model requires $|V|^{n-1}C_n$ number of parameters. However, since $|V| \gg n$, it's approximated as $|V|C_n$ for brevity.

Table 2.2: Types of language models and their respective representations, i.e. $\hat{h}$, of the history

Model	Representation, $\hat{h}$
N -gram	Immediate sequence of $n - 1$ words
Distant-bigram	Distance- k word
Skip-gram	Novel word sequence formed by skipping words
Trigger	Binary occurrences of the correlated words
Bag-of-words	Binary occurrences of all words

Figure 2.1: Types of language models and their respective smoothing paradigms

2.2.1 N -gram Model

N -gram models [Bahl et al., 1983, Nádas, 1984] represent the history as the immediate sequence of $n - 1$ words. Other words beyond the n -gram context are truncated to alleviate the data scarcity problem.

$$P(t = w_i | h = w_1^{i-1}) \approx P_{\text{NG}}(t = w_i | h_{\text{NG}} = w_{i-n+1}^{i-1}) \quad (2.8)$$

where h_{NG} denotes the reduced representation of h as considered by the n -gram models. The number of parameters required by the model is hence limited to $|V|^n \ll |V|^{i-1}$.

Figure 2.2 shows an example where a fourgram model, i.e. $n=4$, is used to estimate

the probability of the word “*market*” based on the immediate three-word sequence “*rest of the*” in the history.

Figure 2.2: A fourgram model represents the history as the immediate three-word sequence.

Smoothing methods for n -gram modeling can be vaguely classified into: discounting, clustering, and neural networks. These methods will be discussed in the following subsections.

Discounting

The idea of discounting is to redistribute the count mass in the data among all possible n -grams, such that the count mass of the seen events would be discounted and reassigned to the unseen events. Particularly, the distribution obtained by maximum likelihood (ML) estimation, as shown in Equation (2.9), is smoothed to avoid any unseen n -gram being assigned zero-probability. ML estimate is unsuitable to be used for language modeling as any unseen n -gram would result the respective word sequence to be assigned a zero-probability, see Equation (2.6).

$$P_{\text{NG}}(t = w_i | h_{\text{NG}} = w_{i-n+1}^{i-1}) = \frac{\#(h_{\text{NG}} = w_{i-n+1}^{i-1}, t = w_i)}{\sum_{w_j \in V} \#(h_{\text{NG}} = w_{i-n+1}^{i-1}, t = w_j)} \quad (2.9)$$

where $\#(\cdot)$ denotes the count of an event and V denotes the vocabulary.

The zero-probability problem can be avoided by altering the counts of the n -grams, such as by using Laplace smoothing, where each n -gram is assumed to occur once more

than the actual number of times it is observed:

$$\widehat{\#}(\cdot) = \#(\cdot) + 1 \quad (2.10)$$

or Good-Turing smoothing [Good, 1953, Nádas, 1984], which exploits the count-of-counts² statistics of the n -grams:

$$\widehat{\#}(\cdot) = (\#(\cdot) + 1) \frac{\#_+(\#(\cdot) + 1)}{\#_+(\#(\cdot))} \quad (2.11)$$

where $\widehat{\#}(\cdot)$ denotes the smoothed count of $\#(\cdot)$ while $\#_+(\cdot)$ denotes the count-of-count of an event.

Typically, probabilities of n -grams are estimated by considering also the statistics of $(n - 1)$ -grams. The rationale is that lower-order n -grams are generally more frequently seen hence the statistics can be more reliably used for estimation. Specifically, discounted mass is reassigned to a rare or an unseen n -gram as according to how frequent the respective $(n - 1)$ -gram is, e.g. an unseen n -gram with a more frequent $(n - 1)$ -gram would receive more count mass. An n -gram and its respective $(n - 1)$ -gram can be incorporated by two manners: (1) back-off mechanism [Katz, 1987] or (2) linear interpolation [Jelinek and Mercer, 1980].

1. **Back-off:** The back-off mechanism [Katz, 1987] estimates the probability of a rare or an unseen n -gram from its respective $(n - 1)$ -gram probability, i.e. its probability is approximated based on a shorter context in the history. For instance, a trigram model, i.e. $n = 3$, is formulated as follows.

$$\begin{aligned} \widehat{P}_{3G}(t = w_i | h_{3G} = w_{i-2}^{i-1}) \\ = \begin{cases} d_{w_{i-2}^{i-1}} P_{3G}(t = w_i | h_{3G} = w_{i-2}^{i-1}) & \text{if } \#(h_{3G} = w_{i-2}^{i-1}, t = w_i) > K \\ k_{w_{i-2}^{i-1}} \widehat{P}_{2G}(t = w_i | h_{2G} = w_{i-1}) & \text{otherwise} \end{cases} \end{aligned} \quad (2.12)$$

²Count-of-counts is defined as the number of distinct n -grams that contain the same number of count in the data.

where $d_{w_{i-n+1}^{i-1}}$ denotes the discounting factor that determines the amount of probability mass to be deducted from the frequent n -grams while $k_{w_{i-n+1}^{i-1}}$ denotes the back-off weight that controls how the probability mass is assigned to the lower-order n -grams.

2. **Linear interpolation:** The probability of an n -gram can also be linearly interpolated with the probability of its respective $(n-1)$ -gram [Jelinek and Mercer, 1980]. This approach serves a foundation to many other discounting approaches, such as Witten-Bell [Witten and Bell, 1991], Kneser-Ney [Kneser and Ney, 1995], and modified Kneser-Ney [Chen and Goodman, 1996]. For example, a trigram model is formulated as follows.

$$\begin{aligned} \hat{P}_{3G}(t = w_i | h_{3G} = w_{i-2}^{i-1}) \\ = \alpha_{w_{i-2}^{i-1}} P_{3G}(t = w_i | h_{3G} = w_{i-2}^{i-1}) + (1 - \alpha_{w_{i-2}^{i-1}}) \hat{P}_{2G}(t = w_i | h_{2G} = w_{i-1}) \end{aligned} \quad (2.13)$$

where $\alpha_{w_{i-2}^{i-1}}$ denotes the interpolation weight. Different from the back-off mechanism, linear interpolation takes the $(n-1)$ -gram probability into account regardless whether the n -gram is unseen or rare.

Clustering

Another strategy to tackle the data scarcity problem is via clustering, where the probability of a given n -gram is computed based on the class of the history. Since the number of classes is typically fewer than the permutations of the history in the order of magnitude, the statistics of the n -gram can be more reliably obtained [Brown et al., 1992, Bahl et al., 1989]. Specifically, as histories are clustered as according to some similarity senses, probabilities of the rare and the unseen n -gram can be estimated from other “similar” n -grams in the same class.

$$P_{NG}(t = w_i | h_{NG} = w_{i-n+1}^{i-1}) = P_{NG}^{\text{class}}(t = w_i | h_{NG}^{\text{class}} = C(w_{i-n+1}^{i-1})) \quad (2.14)$$

where $P_{NG}^{\text{class}}(\cdot | \cdot)$ denotes the n -gram probability estimated based on the class of history, i.e. h_{NG}^{class} , which is mapped by $C(h_{NG}) \rightarrow h_{NG}^{\text{class}}$.

Typically, histories are clustered through a series of questions pertaining the syntactic structure in the context, organized in a decision tree or random forests [Bahl et al., 1989, Xu and Jelinek, 2004, Potamianos and Jelinek, 1998]. The classes of histories can also be defined as the sequence of word-classes in the considered context, where each word is mapped to a word-class based on some syntactic characteristic, such as its part-of-speech (POS) [Brown et al., 1992, Samuelsson and Reichl, 1999, Gao et al., 2001].

$$h_{\text{NG}}^{\text{class}} = C_{\text{w}}(w_{i-n+1})C_{\text{w}}(w_{i-n+2})\dots C_{\text{w}}(w_{i-1}) \quad (2.15)$$

where $C_{\text{w}}(\cdot)$ denotes the mapping function from words to word-classes.

Neural Networks

The idea of using neural networks for smoothing is to represent histories as vectors in the continuous space, such that “similar” histories would reside within a neighboring subspace [Bengio et al., 2003, Schwenk, 2007, Mikolov et al., 2011b, Sundermeyer et al., 2012]. To be more specific, by mapping histories into a lower dimensional space, similarities among histories can be more effectively exploited hence rare or unseen events would provide a “similar” probability as other “similar” events that are more frequently seen in the data. The probability of an n -gram is computed from the representation vector of the history:

$$P_{\text{NG}}(t = w_i | h_{\text{NG}} = w_{i-n+1}^{i-1}) = P_{\text{NG}}^{\text{net}}(t = w_i | \mathbf{h}_{\text{NG}}^{\text{net}} = f(w_{i-n+1}^{i-1})) \quad (2.16)$$

where $f(\cdot)$ denotes the function that maps a discrete context in the history to a continuous vector $\mathbf{h}_{\text{NG}}^{\text{net}}$. As compared to the clustering-based methods, as shown in Equation (2.14), which represent “similar” histories as a common class identifier, neural networks map “similar” contexts in a neighboring subspace providing a more precise estimation.

In the early works, probabilities are estimated by using the feedforward neural networks [Bengio et al., 2003, Schwenk, 2007], where the history is represented as a concatenated vector of each word in the history:

$$\mathbf{h}_{\text{NG}}^{\text{FNN}} = \langle f_{\text{w}}(w_{i-n+1})f_{\text{w}}(w_{i-n+2})\dots f_{\text{w}}(w_{i-1}) \rangle \quad (2.17)$$

where f_w denotes the function that projects a word to a vector, commonly referred to as the word-vector, i.e. $f_w : w_j \rightarrow \mathbf{w}_j$. In more recent works, convolutional neural networks have been used to replace the feed-forward architecture by incorporating pooling and convolution functions in the networks [Dauphin et al., 2016a, Pham et al., 2016a].

Recurrent neural networks [Mikolov et al., 2011b], on the other hand, represent longer context in the history as a concatenation of two vectors: one corresponds to the immediate word while the other one corresponds to the rest of the words in the history:

$$\mathbf{h}_{\text{NG}}^{\text{RNNLM}} = \langle f_c(w_1^{i-2})f_w(w_{i-1}) \rangle \quad (2.18)$$

where f_w denotes the function that projects a word to a word-vector, and f_c denotes the function that computes the context vector, typically implemented by using a latch to store the feedback of the networks. Following a similar idea, the long-short term memory neural networks have been proposed by incorporating gates in the architecture such that the context vector can be more flexibly computed by selectively using words in history [Sundermeyer et al., 2012].

2.2.2 Distant-Bigram Model

Distant-bigram models [Huang et al., 1993, Simons et al., 1997, Klakow, 1998] represent the history by a word located in a specific position. Other words in the history are truncated to alleviate the data scarcity problem, thus a distance- k bigram model can be expressed as follows.

$$P(t = w_i | h = w_1^{i-1}) \approx P_{\text{DBG-}k}(t = w_i | h_{\text{DBG-}k} = w_{i-k}) \quad (2.19)$$

where $h_{\text{DBG-}k}$ denotes the reduced representation of h as considered by the distance- k bigram models. Note that a distance-1 bigram model is equivalent to a bigram model, i.e. an n -gram model with $n = 2$. The number of parameters required by the model is hence limited to $|V|^2 \ll |V|^i$.

Figure 2.3 shows an example where a distance-4 bigram model, i.e. $k=4$, is used to estimate the probability of the word “market” based on the word “the” located in the 4-th position in the history.

Figure 2.3: A distance-4 bigram model represents the history by the word located in the 4-th position.

Similar as the n -gram models, the distant-bigram models have to be smoothed in order to assign more reliable probabilities to the rare or unseen distant bigrams. For the near-distance bigrams, e.g. $k < 3$, Good-Turing [Good, 1953, Nádas, 1984] and Kneser-Ney [Kneser and Ney, 1995] methods are typically used, while for the far-distance bigrams, Dirichlet smoothing [Zhai and Lafferty, 2004] has been shown to be more effective.

As distant-bigram models crudely represent the history by only one word, numerous distant-bigram models of different values of k are usually combined to estimate probabilities more accurately. Combination techniques that are commonly used are linear interpolation [Jelinek and Mercer, 1980], log-linear interpolation [Klakow, 1998] and maximum entropy approach [Rosenfeld, 1996].

2.2.3 Skip-Gram Model

Skip-gram models [Guthrie et al., 2006] represent the history as a novel sequence of words derived from the history. Based on some assumptions or measurements, “irrelevant” words are skipped to alleviate the data scarcity problem. Generally, a skip-gram model

is formulated as follows.

$$P(t = w_i | h = w_1^{i-1}) \approx P_{\text{SKG}}(t = w_i | h_{\text{SKG}} = v_1^n) \quad (2.20)$$

where h_{SKG} denotes the reduced representation of h as considered by the skip-gram model, i.e. v_1^n , a novel sequence of words obtained from w_1^{i-1} . The number of parameters required by the model is hence limited to $|V|^n$, where n denotes the length of the considered word sequence v_1^n .

Figure 2.4 shows an example where a skip-gram model is used to estimate the probability of the word “*market*” based on the novel word sequence “*could be boon for*” derived from the history.

Figure 2.4: A skip-gram model represents the history as a sub-sequence of words in the history.

There have been different methods proposed to implement a skip-gram model, each differs from others based on how the “relevancy” of a word is determined such that the “irrelevant” words are skipped. The variable n -gram models [Siu and Ostendorf, 2000], for example, use the mutual information (MI) score; while for the structured language models [Chelba and Jelinek, 2000, Van Uytsel et al., 2001], the history is parsed such that only the head-words would be considered.

2.2.4 Trigger Model

Trigger models [Lau et al., 1993b, Rosenfeld, 1996] represent the history as a binary value of whether the history contains words that are well-correlated to the given target-word,

referred to as the trigger-words. Occurrence of a trigger-word in the history indicates the respective target-word would be more probably seen. In general, a trigger model can be expressed as follows.

$$P(t = w_i | h = w_1^{i-1}) \approx P_{\text{TGR}}(t = w_i | h_{\text{TGR}} = \{\text{yes, no} | \text{TGR}(w_i) \in \{w_j\}_{j=1}^{i-1}\}) \quad (2.21)$$

where h_{TGR} denotes the reduced representation of h as considered by the trigger models. Function $\text{TGR}(\cdot)$ returns the trigger-word of a given target-word, e.g. $\text{TGR}(\text{"market"}) = \text{"stocks"}$. Correlation between words can be measured with the mutual information (MI) metric [Church and Hanks, 1989].

Figure 2.5 shows an example where a trigger model is used to estimate the probability of the word “*market*” based on the binary occurrence of the word “*stocks*” in the history.

Figure 2.5: A trigger model represents the history as the binary occurrence of the correlated word.

Trigger models are similar to the distant-bigram models, as discussed in Section 2.2.2, in the sense that these two types of model exploit only a single word in the history. However, trigger models disregard the positions of words in the history. Moreover, as the non-correlated words are not considered, there is a vast amount of words that will not be associated to any trigger-word, particularly the function words, e.g. $\text{TGR}(\text{"the"}) = \emptyset$. Hence, a trigger model is typically combined with another model of different type, e.g. an n -gram model, as a mean to provide complementary information about the history.

2.2.5 Bag-of-Words Model

Bag-of-words (BOW) models [Bellegarda, 1998, Mikolov, 2013] represent the history as an unordered list of words. The data scarcity problem is alleviated by disregarding the arrangement of words in the history:

$$P(t = w_i | h = w_1^{i-1}) \approx P_{\text{BOW}}(t = w_i | h_{\text{BOW}} = \{w_j\}_{j=1}^{i-1}) \quad (2.22)$$

where h_{BOW} denotes the reduced representation of h as considered by the BOW model. The number of parameters required by the model is reduced to $|V| \times |V|^{i-2} C_{i-1} \ll |V|^i$.

Figure 2.6 shows an example where a BOW model is used to estimate the probability of the word “*market*” based on the unordered list of words in the history.

Figure 2.6: A BOW model represents the history as an unordered list of words.

Typically, BOW representations are treated as vectors and dimensionality reduction techniques are applied to map the vectors into a lower dimensional space, which the “similarity” among the vectors can be more effectively measured. Latent semantic indexing (LSI) and neural networks are among the commonly used methods for modeling BOW.

Latent Semantic Indexing

The key idea of latent semantic indexing (LSI) is to organize the co-occurrence counts of word-history as a matrix such that by applying singular value decomposition (SVD), both words and histories would be mapped to a low-dimensional space. Specifically, as the co-occurrence counts are typically noisy, i.e. many events are rare or are unseen, SVD

is applied to compute a low-dimensional space in which words and histories can be more reliably represented. Thus, how likely a word is to precede a history, see Equation (2.22), can be computed from the similarity between the respective vectors [Bellegarda, 1998, Coccaro, 2005, Novak and Mammone, 2001].

$$P_{\text{BOW}}(t = w_i | h_{\text{BOW}} = \{w_j\}_{j=1}^{i-1}) = \frac{K_{\text{sim}}(f_w(w_i), f_c(\{w_j\}_{j=1}^{i-1}))}{\sum_{w_j \in V} K_{\text{sim}}(f_w(w_j), f_c(\{w_j\}_{j=1}^{i-1}))} \quad (2.23)$$

where $f_w(\cdot)$ and $f_c(\cdot)$ denote the functions that map words and histories to the low-dimensional space. Function $K_{\text{sim}}(\cdot, \cdot)$ is typically implemented as the cosine similarity.

Typically, the co-occurrence counts are computed by scaling the term-frequency (TF) with the inverse document frequency (IDF): TF refers to the count of a word in a specific history, while IDF refers to the number of times a word occurs in different histories. The motivation of IDF scaling is similar to the MI-based words filtering in the trigger modeling (see Section 2.2.4), i.e. to stress the contribution of the semantically important words. However, different from the trigger models which exploit the binary existences of words in the history, the LSI-based models consider the counts of words.

Other methods used to implement a BOW model include probabilistic-LSI (pLSI) and the latent Dirichlet allocation (LDA). In pLSI, latent variables are introduced to the LSI framework, where words are assumed to possess different distributions with respect to a set of latent variables [Hofmann, 1999]. On the other hand, LDA further assumes these latent variables are generated from a Dirichlet distribution [Blei et al., 2002].

Neural Networks

Similar to the neural networks approach for modeling n -grams (see Section 2.2.1), by projecting BOW vectors into a lower dimensional space, similarity among BOW representations can be more effectively estimated. BOW representations that are “similar” will reside within a neighboring subspace. Hence probabilities estimated based on the “similar” BOW would provide “similar” probability values to a given word [Mikolov, 2013].

$$P_{\text{BOW}}(t = w_i | h_{\text{BOW}} = \{w_j\}_{j=1}^{i-1}) = P_{\text{BOW}}^{\text{net}}(t = w_i | \mathbf{h}_{\text{BOW}} = f(\{w_j\}_{j=1}^{i-1})) \quad (2.24)$$

where $f(\cdot)$ denotes the function that maps a BOW to a continuous vector $\mathbf{h}_{\text{BOW}}$.

2.3 Model Combination

Language models of different types are combined to incorporate complementary representations of the history to estimate probabilities more accurately. As each representation is devoted to capture some specific information about the history, model combination allows for a richer information to be gathered resulting into improved language models. Particularly, the data scarcity problem of modeling the entire history, see Equation (2.6), is tackled by first reducing the history into multiple representations, then the respective models are combined. Say a history h is considered as a union of representations of h_A and h_B , model combination computes the probabilities as follows.

$$P(t = w_i | h) \approx P_{A-B}(t = w_i | h_A, h_B) \approx f_{\text{combine}}(P_A(t = w_i | h_A), P_B(t = w_i | h_B)) \quad (2.25)$$

where $f_{\text{combine}}(\cdot, \cdot)$ denotes the combination function. The complexity in building P_{A-B} hence will not be more severe than building any of the models of P_A and P_B .

The advantage of model combination can be depicted by an example of combining an n -gram model with a BOW model. As the n -gram models are effective in exploiting the near context but the performance is limited when longer context is considered; while the BOW models are feasible in exploiting the long context but neglect the crucial information about words arrangement in the near context – these two types of model complement each other very well.

$$P(t = w_i | h = w_1^{i-1}) \approx P_{\text{NG-BOW}}(t = w_i | h_{\text{NG}} = w_{i-n+1}^{i-1}, h_{\text{BOW}} = \{w_j\}_{j=1}^{i-1}) \quad (2.26)$$

Consider a fourgram model is used to estimate the probability of the word “*market*” in the word sequence below, semantic important words such as “*collapse*” and “*stocks*” are beyond the reach of the n -gram model.

a collapse in blue chip stocks could be a boon for the rest of the market

Figure 2.7 illustrates an example of combination of the fourgram model with a BOW model, which the long context in the history can be exploited to estimate the probability of the word “market”.

Figure 2.7: A fourgram model is combined with a BOW model to exploit longer context in the history to estimate probability of the word “market”.

In general, language models can be combined through: linear interpolation (LI), log-linear interpolation (LLI), and maximum entropy (ME). These three methods are discussed in more detail in the following sub-sections.

2.3.1 Linear Interpolation

Linear interpolation (LI) computes the linear average of the probabilities provided by the models [Jelinek and Mercer, 1980].

$$P_{A-B}(t = w_i | h_A, h_B) = \alpha_A P_A(t = w_i | h_A) + \alpha_B P_B(t = w_i | h_B) \quad (2.27)$$

where h_A and h_B denote two different types of representation of the history. The weights α_A and α_B control the contribution of each model in the combination, where $\alpha_A + \alpha_B = 1$ to ensure the probabilities add up to one.

LI can be generalized by applying weights that vary across different contexts of history [Bacchiani et al., 2006, Bo-June Hsu, 2007, Liu et al., 2013]. The motive is that as

each model is effective in exploiting certain contexts, e.g. contexts that consist of semantic important words are preferable by the BOW models, a history-dependent weighting scheme allows the strengths in the models to be better fused.

$$P_{A-B}(t = w_i|h_A, h_B) = \alpha_A(h_A, h_B)P_A(t = w_i|h_A) + \alpha_B(h_A, h_B)P_B(t = w_i|h_B) \quad (2.28)$$

where $\alpha_A(h_A, h_B)$ and $\alpha_B(h_A, h_B)$ denote the history-dependent weights. Due to the large amount of free parameters, the weights are typically clustered based on the permutations in the context [Bahl et al., 1983, Chen and Goodman, 1996, Zhou et al., 2002].

2.3.2 Log-Linear Interpolation

Log-linear interpolation (LLI), on another hand, computes the geometric average of the probabilities provided by the models [Klakow, 1998, Heafield et al., 2016].

$$P_{A-B}(t = w_i|h_A, h_B) = \frac{P_A(t = w_i|h_A)^{\alpha_A} P_B(t = w_i|h_B)^{\alpha_B}}{Z(h_A, h_B)} \quad (2.29)$$

where $Z(h_A, h_B)$ denotes the normalization factor that ensures the probabilities add up to one. The weights α_A and α_B determine the contribution of each model in the combination. Generally, LLI can provide better generalization than LI – in the formal derivation of LLI, the weights are optimized to minimize the distance of the probabilities from the uniform distribution [Gutkin, 2000].

Similar to LI, history-dependent weights can be applied for a more complex setting of LLI:

$$P_{A-B}(t = w_i|h_A, h_B) = \frac{P_A(t = w_i|h_A)^{\alpha_A(h_A, h_B)} P_B(t = w_i|h_B)^{\alpha_B(h_A, h_B)}}{Z(h_A, h_B)} \quad (2.30)$$

where $\alpha_A(h_A, h_B)$ and $\alpha_B(h_A, h_B)$ denote the history-dependent weights.

2.3.3 Maximum Entropy

The principle of maximum entropy (ME) considers instances of events as binary features and the probability is in the form of [Lau et al., 1993a, Rosenfeld, 1996]:

$$P_{A-B}(t = w_i | h_A, h_B) = \prod_j \mu_j^{f_j(h_A, t=w_i)} \prod_k \mu_k^{f_k(h_B, t=w_i)} \quad (2.31)$$

where $f_j(h_A, t = w_i)$ and $f_k(h_B, t = w_i)$ are referred to as the constraint functions which correspond to the j -th and the k -th feature, respectively. Parameters μ_j and μ_k are constants obtained typically through generalized iterative scaling (GIS) algorithm.

Typically, a constraint function returns 1 if the considered feature is seen, else 0. For example, a feature of fourgram “*rest of the market*” is expressed as follows:

$$f(h_{4G} = \text{“rest of the”}, t = \text{“market”}) = \begin{cases} 1 & \text{if } h_{4G} = \text{“rest of the” and } t = \text{“market”} \\ 0 & \text{otherwise} \end{cases} \quad (2.32)$$

ME can be seen as a generalization of LLI, where LLI adapts only the weights to minimize the distance of the combined distribution to the uniform distribution; while ME adapts the entire parameters in the model.

2.4 The Inconsistency Problem

Model combination through LI and LLI are straightforward and easy to be implemented, but inter-dependencies among the representations of the history are typically neglected. Such problem is due to the fact that each representation is modeled individually, and the final model is computed through averaging the probabilities, which takes no measure to consider the inter-dependencies. As a consequence, the parameters in the model are not consistent to the statistics of the events as observed in the data.

$$P_{A-B}(\cdot | h_A, h_B) = \alpha_A P_A(\cdot | h_A) + \alpha_B P_B(\cdot | h_B) \neq \frac{\#(h_A, h_B, \cdot)}{\#(h_A, h_B)} \quad (2.33)$$

where h_A and h_B denote two different types of representations of the history. There will be events over-counted or under-counted. Although by carefully fine-tuning the interpolation weights, particularly in the history-dependent weighting scheme, see Equations (2.28) and

(2.30), the consistency can be recovered, such methods lack theoretical background and their corresponding computations are non-trivial.

For the case of ME, although the duality with the solution of ML implies that multiple representations can be exploited jointly [Rosenfeld, 1996], as the instances obtained from the data are treated as binary features, it is suitable only for exploiting simple events, such as n -gram, distant-bigram, and trigger models. For BOW that is effective in exploiting the long context, many instances would be unseen, i.e. it is unlikely to observe an exact same BOW from two sets of data. As a result, many of the features trained in the ME models cannot be utilized. Moreover, building a ME model is computational intensive.

2.5 Summary

This chapter presented an overview of language modeling, particularly the data scarcity problem in building a language model along with different approaches that have been proposed to address it. In general, these approaches estimate probabilities of word sequences based on features that are more susceptible to the data scarcity problem. Combining language models aims at incorporating representations of multiple types to estimate probabilities more accurately, but the commonly used combination techniques such as LI and LLI neglect the inter-dependencies among the features. Next chapter will provide a further discussion on this problem.

In Section 2.1, language models have been discussed, including the definition, applications, and evaluation. The data scarcity problem in building a language model has been addressed in Section 2.2. It has been shown how the probability of a word sequence can be computed based on the conditional probabilities of a word given its previous history. A language model can be classified into categories based on how the history is represented: n -gram, distant-bigram, skip-gram, trigger, and bag-of-words, each constitutes a sub-section from 2.2.1 to 2.2.5.

In Section 2.3, model combination techniques such as linear interpolation, log-linear

interpolation, and maximum entropy have been discussed. The weakness of these techniques in neglecting the inter-dependencies between the representations has been addressed and further elaboration will be given in next chapter.

Chapter 3

Term-Distance Term-Occurrence Language Model

One way to improve the performance of a language model is to exploit longer context in the history. Typically, a long context is represented as a combination of several simple events, such as a sub-sequence of words and a bag-of-words. However, since each event is modeled separately, the inter-dependencies will be neglected by the model. As a result, information captured from the long context cannot be optimally utilized.

In this chapter, a novel language model, coined as the term-distance term-occurrence (TDTO) model, is proposed. The TDTO model exploits the long context based on the term-distance (TD) and term-occurrence (TO) events that correspond to the positions and occurrences of words in the context. By jointly modeling these two types of events, the TDTO model attempts to capture the inter-dependencies, as a better way to utilize information about the long context. Also, by interpreting the TDTO model in terms of a log-linear interpolation framework, the impacts of the TD and TO components can be tuned to fit the TDTO model to languages of different styles.

Section 3.1 provides the background and motivation of the TDTO modeling, i.e. the conventional approaches to exploiting the long context tend to neglect the inter-dependencies between the considered events. In Section 3.2, the details of the TDTO

modeling, including the derivation and the estimation of the TD and TO model components, is presented. Section 3.3 then summarizes this chapter.

The evaluation of the TDTO model will be presented in the next chapter.

3.1 Background

The probability of a word sequence, say $s = (w_1, w_2, \dots, w_{m-1}, w_m) = w_1^m$, can be computed by using the chain rule, as follows.

$$P(s = w_1^m) = \prod_{i=1}^m P(t = w_i | h = w_1^{i-1}) \quad (3.1)$$

where t denotes an arbitrary target-word w_i in the sequence and h denotes the history. Due to the data scarcity problem, the word sequence in the history, i.e. w_1^{i-1} , is typically reduced to representations that consist of simple events such as sub-sequence of words or bag-of-words (BOW).

3.1.1 Model Combination

The performance of a language model can be improved by combining multiple representations of the history such that richer information can be captured. Typically, sub-sequence of words in the immediate context, as exploited by the n -gram models, is combined with another representation that spans longer context, for example BOW.

$$P(t = w_i | h = w_1^{i-1}) \approx P_{\text{NG-BOW}}(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1}, h_{\text{BOW}} = \{w\}_{i-n_{\text{BOW}}+1}^{i-1}) \quad (3.2)$$

where h_{NG} and h_{BOW} denote the $(n_{\text{NG}} - 1)$ word sequence and the BOW in the history, respectively. The BOW is obtained from a context length of n_{BOW} , which $n_{\text{BOW}} \gg n_{\text{NG}}$.

In general, each representation is modeled separately and the models are combined through linear interpolation (LI) [Jelinek and Mercer, 1980] or log-linear interpolation (LLI) [Klakow, 1998], as discussed in Section 2.3.1 and 2.3.2. LI and LLI provide an easy

and cost effective manner for model combination. For example, LLI of an n -gram model with a BOW model can be computed as the geometric average, as follows:

$$P_{\text{NG-BOW}}^{\text{LLI}}(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1}, h_{\text{BOW}} = w_{i-n_{\text{BOW}}+1}^{i-1}) \approx \frac{P_{\text{NG}}(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1})^{\beta_{\text{NG}}} P_{\text{BOW}}(t = w_i | h_{\text{BOW}} = \{w\}_{i-n_{\text{BOW}}+1}^{i-1})^{\beta_{\text{BOW}}}}{Z(h_{\text{NG}}, h_{\text{BOW}})} \quad (3.3)$$

where $P_{\text{NG}}(\cdot|\cdot)$ and $P_{\text{BOW}}(\cdot|\cdot)$ denote the n -gram and the BOW component models, respectively, while $P_{\text{NG-BOW}}^{\text{LLI}}(\cdot|\cdot)$ denotes the LLI combined model. The impact of each component model towards to the final estimation is controlled by the interpolation weights, i.e. β_{NG} and β_{BOW} .

In the formalism of LI and LLI, the data scarcity problem is tackled by assuming the considered representations to be independent from each other¹, hence each source of information can be exploited separately by using different models. To be more specific, LI and LLI limit the complexity in the combined model to not exceed the complexity incurred by each of the component model in separate.

3.1.2 Parameters Inconsistency

As mentioned in Section 2.4, LI and LLI tackle the data scarcity problem by relaxing the inter-dependencies among the representations. As a consequence, the parameters in the combined model will not be consistent to the statistics in the given training data. Particularly, in the LLI of an n -gram model with a BOW model, see Equation (3.3), as words in the immediate context are commonly exploited by both representations – there will be events over-counted or under-counted in the combined model, shown as follows.

$$P_{\text{NG-BOW}}^{\text{LLI}}(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1}, h_{\text{BOW}} = w_{i-n_{\text{BOW}}+1}^{i-1}) \neq \frac{\#(h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1}, h_{\text{BOW}} = w_{i-n_{\text{BOW}}+1}^{i-1}, t = w_i)}{\#(h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1}, h_{\text{BOW}} = w_{i-n_{\text{BOW}}+1}^{i-1})} \quad (3.4)$$

¹More specifically, conditional independent given the target-word.

where $\#(\cdot)$ denotes the count of an event in the training data. Combining other types of language models by using LI or LLI would suffer from the same problem, e.g. n -gram with distant-bigram, skip-gram or trigger.

The problem of parameter inconsistency has been addressed in the works of maximum entropy (ME) language modeling [Lau et al., 1993a, Lau et al., 1993b, Rosenfeld, 1996]. Particularly, the duality of maximum entropy and maximum likelihood implies that the inter-dependencies between the representations are captured in the models. However, learning a ME model is computationally intensive. Also, a proper way to exploit BOW by using a ME model is yet to be explored.

Figure 3.1: Inter-dependencies between n -gram and BOW are not captured by LI and LLI.

Next section proposes an approach to exploiting the *joint event* of the representations such that the inter-dependencies can be captured. That is to say, the probability of a given word is estimated based on less assumptions made on the word sequence in the history.

3.2 Term-Distance and Term-Occurrence

In order to tackle the parameters inconsistency problem, as discussed in Section 3.1.2, an approach to joint modeling multiple representations of the history is proposed. Different from the LI or the LLI, the inter-dependencies among the representations are attempted to be captured. Assume the history is exploited based on events of different specificity, the data scarcity problem can be alleviated by *decoupling* the model into multiple compo-

nents, such that those components that consider grosser events would be able to provide more reliable probabilities.

In this investigation, two types of representations are considered:

1. **Sub-sequence of words:** The n -gram models [Bahl et al., 1983, Nádas, 1984] exploit the sub-sequence of words in the immediate context to estimate probabilities. The idea is to capture the syntactic information that is crucial for word prediction. However, due to the data scarcity problem, the order of an n -gram is usually fixed to be low, e.g. $n = 3$, then combining it with another representation that exploits longer context is necessary.
2. **Bag-of-words (BOW):** The BOW models [Bellegarda, 1998, Mikolov, 2013] are capable to exploit long context without drastically increasing the data scarcity problem. A BOW is effective for capturing semantic information by considering only the binary occurrences of words. It is typically combined with n -gram for language modeling.

The details of the proposed model will be elaborated in the following sub-sections.

3.2.1 Joint Modeling of N -gram and BOW

To jointly model the n -gram and BOW, the formulation begins by expressing the probability function in Equation (3.2) in terms of prior and likelihood by using Bayes' rule:

$$\begin{aligned}
 P_{\text{NG-BOW}}(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1}, h_{\text{BOW}} = \{w\}_{i-n_{\text{BOW}}+1}^{i-1}) \\
 = \frac{P(t = w_i)P(h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1}, h_{\text{BOW}} = \{w\}_{i-n_{\text{BOW}}+1}^{i-1} | t = w_i)}{Z(h_{\text{NG}}, h_{\text{BOW}})} \quad (3.5)
 \end{aligned}$$

where h_{NG} and h_{BOW} denote the $(n_{\text{NG}} - 1)$ sub-sequence of words and BOW in the history, respectively. The BOW is obtained from a context length of n_{BOW} , which $n_{\text{BOW}} \gg n_{\text{NG}}$. The normalization term $Z(h_{\text{NG}}, h_{\text{BOW}})$ ensures the probabilities add up to one.

In order to estimate probabilities in a more feasible manner, by using the chain rule, the likelihood of the joint of n -gram and BOW, see Equation (3.5), is *decoupled* into two separate components:

$$P_{\text{NG-BOW}}^{\text{LLI}}\left(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1}, h_{\text{BOW}} = \{w\}_{i-n_{\text{BOW}}+1}^{i-1}\right) = \frac{\begin{pmatrix} P(t = w_i) \\ P\left(h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1} | h_{\text{BOW}} = \{w\}_{i-n_{\text{NG}}+1}^{i-1}, t = w_i\right) \\ P\left(h_{\text{BOW}} = \{w\}_{i-n_{\text{BOW}}+1}^{i-1} | t = w_i\right) \end{pmatrix}}{Z(h_{\text{NG}}, h_{\text{BOW}})} \quad (3.6)$$

where

- $P\left(h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1} | h_{\text{BOW}} = \{w\}_{i-n_{\text{NG}}+1}^{i-1}, t = w_i\right)$ denotes the likelihood of the n -gram, conditioned on the BOW of the respective context. By conditioning the word sequence $w_{i-n_{\text{NG}}+1}^{i-1}$ on its respective BOW $\{w\}_{i-n_{\text{NG}}+1}^{i-1}$, the inter-dependency between these two representations is captured.
- $P\left(h_{\text{BOW}} = \{w\}_{i-n_{\text{BOW}}+1}^{i-1} | t = w_i\right)$ denotes the likelihood of the BOW.

Decoupling provides a means to tackle the data scarcity problem, particularly in the combined model which demands $|V|C_{n_{\text{BOW}}} \times |V|^{n_{\text{NG}}}$ number of parameters. By computing the likelihood of the joint event by using two terms, the n -gram likelihood function still demands the same amount of parameters, but the number of parameters required by the BOW likelihood function is reduced to only $|V|C_{n_{\text{BOW}}}$. Under the situation of a rare event in the training data, the BOW likelihood would provide a more reliable probability as compensation to the overall estimation.

3.2.2 Modeling Distance and Occurrence of Word-Pairs

Decoupling yields a more feasible formulation to compute probabilities based on n -gram and BOW, however learning the likelihood functions, especially the n -gram given BOW likelihood, see Equation (3.6), is non-trivial. To ease the implementation, words in the

history are assumed to be conditionally independent given the target-word, i.e. $w_{i-j} \perp\!\!\!\perp w_{i-k}|w_i$ where $j \neq k$. Hence the probability function in Equation (3.6) can be reduced as:

$$P_{\text{NG-BOW}}^{\text{LLI}}\left(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1}, h_{\text{BOW}} = \{w\}_{i-n_{\text{BOW}}+1}^{i-1}\right) \\ = \frac{\left(\begin{array}{c} P(t = w_i) \\ \prod_{k=1}^{i-n_{\text{NG}}+1} P\left(h_{\text{NG}-k} = w_{i-k} | h_{\text{BOW}-k} = \{w\}_{i-k}, t = w_i\right) \\ \prod_{k=1}^{i-n_{\text{BOW}}+1} P\left(h_{\text{BOW}-k} = \{w\}_{i-k} | t = w_i\right) \end{array} \right)}{Z(h_{\text{NG}-1}, h_{\text{NG}-2}, \dots, h_{\text{BOW}-1}, h_{\text{BOW}-2}, \dots)} \quad (3.7)$$

where $h_{\text{NG}-k}$ denotes k -th word in the history, while $h_{\text{BOW}-k}$ denotes the binary occurrence of k -th word within the history, i.e. $\bigcup_{k=1}^{n_{\text{NG}}-1} h_{\text{NG}-k} = h_{\text{NG}}$ and $\bigcup_{k=1}^{n_{\text{BOW}}-1} h_{\text{BOW}-k} = h_{\text{BOW}}$. These two types of events are used to approximate the n -gram and BOW, where the former considers the arrangement of words while the latter considers only the binary occurrences of words.

The formulation in Equation (3.7) provides an approach to compute the probability of target-word w_i based on the distance and the occurrence of word-pair (w_{i-k}, w_i) , for $1 \leq k \leq n_{\text{NG}}, n_{\text{BOW}}$.

Let's define:

- $\Delta(w_j)$ as the discrete distance of w_j in the history from a given target-word, referred to as the *term-distance* (TD). Hence $(\Delta(w_{i-k}), w_i)$ denotes the distance in the word-pair (w_{i-k}, w_i) .
- $O(w_j)$ as the binary occurrence of w_j in the history of a given target-word, referred to as the *term-occurrence* (TO). Hence $(O(w_{i-k}), w_i)$ denotes the occurrence of the word-pair (w_{i-k}, w_i) .

By expressing the formulation in Equation (3.7) in terms of TD and TO, the model is hence referred to as the term-distance term-occurrence (TDTO) model:

$$\begin{aligned}
P_{\text{TDTO}}(t = w_i | h_{\text{TDTO}} = w_{i-n+1}^{i-1}) \\
= \frac{\left(\frac{P(t = w_i) \prod_{k=1}^{n_{\text{TD}}} P_{\text{TD}-k}(h_{\text{TD}-k} = \Delta(w_{i-k}) | h_{\text{TO}-k} = \text{O}(w_{i-k}), t = w_i)}{\prod_{k=1}^{n_{\text{TO}}} P_{\text{TO}-k}(h_{\text{TO}-k} = \text{O}(w_{i-k}) | t = w_i)} \right)}{Z(h_{\text{TD}-1}, h_{\text{TD}-2}, \dots, h_{\text{TO}-1}, h_{\text{TO}-2}, \dots)} \quad (3.8)
\end{aligned}$$

where n denotes the length of the history considered by the model, i.e. $n = \max(n_{\text{TD}}, n_{\text{TO}})$.

The formulation in Equation (3.8) consists of three different model components, which are referred to as:

- The prior, $P(t = w_i)$, is the initial belief that the target-word under consideration would occur. It may encompass expert knowledge, or sampled statistics from the training data.
- The term-distance (TD) likelihoods, $P_{\text{TD}-k}(h_{\text{TD}-k} = \Delta(w_{i-k}) | h_{\text{TO}} = \text{O}(w_{i-k}), t = w_i)$, for $k = 1, 2, \dots, n_{\text{TD}}$ measure how likely a distance of k will be encountered in between the history words w_{i-k} and the target-word w_i .
- The term-occurrence (TO) likelihoods, $P_{\text{TO}-k}(h = \text{O}(w_{i-k}) | t = w_i)$, for $k = 1, 2, \dots, n_{\text{TO}}$ measure how likely words w_{i-k} will occur within the history-context (regardless its position) of target-word w_i .

The proposed framework provides an alternative mechanism for estimating probabilities by expressing a word-history event in terms of the distances and the occurrences of word-pairs encountered in a considered context. This formulation allows for a more principled characterization of running words in language, especially when modeling long context in the history. Within this framework, actual probabilities for distant relevant words can be taken into account without backing-off due to the occurrence of a rare word within the n -gram context.

Besides capturing the inter-dependency between the TD and TO events, the proposed framework offers an additional flexibility by exploiting and modeling the distance and occurrence components in a separated way. Such convenience will be discussed in more detail in the following sub-sections.

3.2.3 Model Generalization

Following the formulation presented in the previous sub-section, after presenting and describing how a context can be decomposed into three model components: prior, TD and TO, this sub-section elaborates how this decoupling formulation can be further exploited to gain more flexibility in the TDTO model. Firstly, some of the inter-dependencies among words in the history are restored by introducing an n -gram component to function as a prior. Secondly, a log-linear weighting scheme is applied to allow for model tuning and adaptation.

Conditional Independence Restoration

The likelihood of the n -gram, i.e. $P(h = w_{i-n+1}^{i-1} | t = w_i)$, is augmented to restore some of the inter-dependencies among the history-words, which have been discarded by the conditional independence assumption in earlier derivation, see Equation (3.7). Since the n -gram is essentially the raw instance of the word sequence, this step is important to compensate the information loss resulting from the independence assumption; especially in the near context, for which good n -gram probabilities can be estimated. In this way, the idea is to combine the n -gram likelihood with a more informed prior to yield a better approximation of the target-word posterior.

Generally, the n -gram component considers a shorter context as compared to both the TD and the TO model components. The n -gram's history is denoted to be h_{NG} , of length n_{NG} , and the TD and the TO's history-context to be h_{TD} and h_{TO} , of length of n_{TD} and n_{TO} , respectively, where n_{TD} and n_{TO} are typically greater than n_{NG} . The formulation of the proposed model by means of the three components explained above is

thus expressed as follows:

$$\begin{aligned}
P_{\text{TDTO}}(t = w_i | h_{\text{TDTO}} = w_{i-n+1}^{i-1}) \\
= \frac{\left(\frac{P_{\text{NG}}(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1})}{\prod_{k=1}^{n_{\text{TD}}-1} P_{\text{TD}-k}(h_{\text{TD}-k} = \Delta(w_{i-k}) | h_{\text{TO}-k} = O(w_{i-k}), t = w_i)} \right)}{Z(h_{\text{NG}}, h_{\text{TD}-1}, h_{\text{TD}-2}, \dots, h_{\text{TO}-1}, h_{\text{TO}-2}, \dots)} \quad (3.9)
\end{aligned}$$

where the overall length of the considered history n is usually chosen as n_{TD} or n_{TO} , i.e. the longer context. Note that although the TD component is derived from the n -gram, see Equation (3.7), it is capable to exploit longer context, i.e. $n_{\text{TD}} > n_{\text{NG}}$. Since words in the history are assumed conditional independent from each other, an unseen event would not have to be backed-off to a shorter context as compared to the conventional n -gram models.

Component models weighting

As the TD and TO components capture complementary information, each component can be applied independently of the other. This offers a convenience for generalization in which the overall model is intended to emphasize either the distance or the occurrence information. This can be realized through assigning different log-linear weights to each of the model components with the aim of adjusting their contributions towards the final estimation:

$$\begin{aligned}
P_{\text{TDTO}}(t = w_i | h_{\text{TDTO}} = w_{i-n+1}^{i-1}) \\
= \frac{\left(\frac{P_{\text{NG}}(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1})^{\beta_{\text{NG}}}}{\prod_{k=1}^{n_{\text{TD}}-1} P_{\text{TD}-k}(h_{\text{TD}-k} = \Delta(w_{i-k}) | h_{\text{TO}-k} = O(w_{i-k}), t = w_i)^{\beta_{\text{TD}}}} \right)}{Z(h_{\text{NG}}, h_{\text{TD}-1}, h_{\text{TD}-2}, \dots, h_{\text{TO}-1}, h_{\text{TO}-2}, \dots, \beta_{\text{NG}}, \beta_{\text{TD}}, \beta_{\text{TO}})} \quad (3.10)
\end{aligned}$$

where the weights: β_{NG} , β_{TD} , and β_{TO} , can be experimentally set to fit the application.

One benefit of the generalization in Equation (3.10) is that it offers a good compromise between the distant-bigram model and the trigger model. The distant-bigram model uses the coupled word-pair, in which distance and occurrence components are assigned an equal weight, but the model is not effective in capturing far context. While the trigger model is capable of exploring context farther behind, but the distance component is entirely ignored. By using a weighting scheme, as shown in Equation (3.10), the distance information is accounted for in a more structured manner, which allows for better fitting the model to the data. The out-performance of the proposed model over the distant-bigram and the trigger models will be empirically shown in next chapter.

The weighting scheme can be manually tailored or empirically optimized for a specific application given some data evidence. For example, to identify text in which TD information is more discriminative, the weights of the n -gram and TO components can be tuned to zero. In such way, the targeted text can be better distinguished from text belonging to other competing classes. In the worst scenario, the model can be always reduced back to an n -gram model, by fixing β_{TD} and β_{TO} to zero, hence becoming a standard n -gram model text classifier.

The formulation depicted in Equation (3.10) is essentially a log-linear interpolated model of the n -gram, TD, and TO components. In the discussion throughout this thesis, the model in Equation (3.10) will be referred to as TDTO model.

3.2.4 Model Component Computation

This sub-section proposes methodologies for computing probability estimates of the TD and the TO components from word-pair counts. As mentioned previously, the TD component considers only the distance information, regardless of the actual count of the word-pair. On the other hand, the TO component accounts for the count information of the word-pair, regardless of the actual distance between the words. Computing the likelihoods for the TD and the TO components from the word-pair counts requires the

maximum likelihood estimates to be smoothed to avoid zero-probability from being given by the unseen events.

Term distance

The TD likelihood measures how likely a distance would be observed between a history-word and a target-word. The TD likelihood for a distance of k can be computed from the empirical counts, denoted by $\#(\cdot)$, as follows:

$$P_{\text{TD}-k}(h_{\text{TD}-k} = \Delta(w_{i-k}) | h_{\text{TO}-k} = \text{O}(w_{i-k}), t = w_i) \\ = \frac{\#(h_{\text{TD}-k} = \Delta(w_{i-k}), h_{\text{TO}-k} = \text{O}(w_{i-k}), t = w_i)}{\#(h_{\text{TO}-k} = \text{O}(w_{i-k}), t = w_i)} \quad (3.11)$$

Note that the above TD likelihood is computed by normalizing the count of a word-pair separated by distance of k with respect to the total counts of the same word-pair at any distance $1 \leq k \leq n_{\text{TD}}$, within the considered context. Hence, the count information, i.e. how frequent the word-pair occurs, is ignored and the probability only accounts for relative position information.

As any other empirical probability estimate, the TD likelihood requires the use of smoothing techniques for resolving unseen events. Particularly, smoothing the TD likelihood requires dealing with two main problems. First, a word-pair with distance of k has zero count in the training set: $\#(h_{\text{TD}-k} = \Delta(w_{i-k}), h_{\text{TO}-k} = \text{O}(w_{i-k}), t = w_i) = 0$, which will result in a null numerator in Equation (3.11). Second, a word-pair is never seen at any distance, within the history-context window, in the training set: $\#(h_{\text{TO}-k} = \text{O}(w_{i-k}), t = w_i) = 0$, which will result in a divide-by-zero problem in Equation (3.11).

In order to tackle the first problem, the counts of the word-pairs with distance k , have to be redistributed to assign some probability mass to the unseen events. This method is applicable only if the total count is not zero, i.e. $\#(h_{\text{TD}-k} = \Delta(w_{i-k}), h_{\text{TO}-k} = \text{O}(w_{i-k}), t = w_i) \neq 0$.

Notice that the assumption is made that the counts of the word-pairs are smooth

along contiguous distance position. In other words, the distribution of $\#(h_{\text{TD}-k} = \Delta(w_{i-k}), h_{\text{TO}-k} = O(w_{i-k}), t = w_i)$ is assumed to vary smoothly with distance. This assumption further suggests that a word-pair with distance of k has certain effect over the occurrence of the same word-pair with different distances. Such assumption has also been empirically shown [Beeferman et al., 1997]. Based on this assumption, a moving average filter is proposed for smoothing the TD model counts:

$$\begin{aligned} \widehat{\#}(h_{\text{TD}-k} = \Delta(w_{i-k}), h_{\text{TO}-k} = O(w_{i-k}), t = w_i) \\ = \sum_{l=1}^{n_{\text{TD}}} \mu_l \#(h_{\text{TD}-l} = \Delta(w_{i-l}), h_{\text{TD}-k} = O(w_{i-k}), t = w_i) \end{aligned} \quad (3.12)$$

where μ_l is a weighting factor based on distance l .

To highlight the gradual reduction of the additive count when a word-pair distance gets farther from k , the weights of the moving average filter can be defined as follows:

$$\mu_l = \begin{cases} \alpha^{|l-k|} & \text{for } l \neq k \\ 1 - \sum_{k'=1, k' \neq l}^{n_T-1} \alpha^{|l-k'|} & \text{for } l = k \end{cases} \quad (3.13)$$

where α is a decay factor, which can be adjusted by evaluating the model performance over a development-set, or arbitrarily fixed to a reasonable value, for instance 0.1. Notice that this strategy is different from other conventional smoothing techniques, which mainly rely on count-of-counts statistics to re-estimate the counts. Such conventional smoothing methods are not applicable here due to the different nature of the TD model, which focuses on word positions rather than on occurrences.

For the second problem, in which a word-pair is never seen at any distance within the history-context window, the observation window is assumed to not being wide enough and such a word-pair would be observed at a length that is greater than $n_{\text{TD}} - 1$, eventually. If there is at least one occurrence of the word-pair at a distance of $m > n_{\text{TD}} - 1$ within the training set, the corresponding smoothed count, as computed by the moving average filter as depicted in Equation (3.12), would be as follows:

$$\hat{\#}(h = \Delta(w_{i-k}), h = O(w_{i-k}), t = w_i) = \mu_m \#(h_{\text{ext}} = \Delta(w_{i-m}), h = O(w_{i-k}), t = w_i) \quad (3.14)$$

where h_{ext} denotes the extended history-context of length m , which is merely wide enough to observe an occurrence of the word-pair under consideration. The coefficient μ_m denotes the proportion of the word-pair count in the extended context h_{ext} contributing to the smoothing.

Term occurrence

As a complement to the TD component, the TO model pays attention only to the count of word-pairs, regardless of their separation distance, for all distances within the history-context window with $1 < k < n_{\text{TO}} - 1$. According to this, the TO likelihood measures how likely a history-word would precede a given target-word within the considered history-context, regardless of its position. This probability can be computed from the empirical counts as follows:

$$P_{\text{TO}-k}(h = O(w_{i-k})|t = w_i) = \frac{\#(h = O(w_{i-k}), t = w_i)}{\#(t = w_i)} \quad (3.15)$$

Note that the likelihood in Equation (3.15) is computed by accumulating the counts of the word-pair for all distances within the considered history-context window. Hence, the distance information is ignored and the probability only accounts for the co-occurrence counts.

One advantage of considering word occurrences regardless of their relative positions is that the sparseness problem becomes less severe. For instance, given an observation window that is long enough, it is more likely to observe a history-word in the training set than in the case of only inspecting one particular position in the history-context.

In fact, the TO model is very similar to the trigger language model, as both models attempt to predict the target-word (triggered word) based on the presence of a history-word (the trigger). However, differently from the trigger language model, the TO model

takes into account all possible word-pairs, without filtering out those that are weakly associated. Ideally, in the TO model formulation, the weak association between word-pairs shall be reflected by the model probabilities. Additionally, the proposed TO probability is computed by taking into account the actual number of occurrences of the history-word within the context, whereas in the trigger language model, the probability is computed by considering only the number of history-contexts that contain the trigger, i.e. when the trigger word occurs more than once in the history-context, it is counted as one occurrence only.

In order to avoid zero probability estimates when $\#(h = O(w_{i-k}), t = w_i) = 0$, the TO likelihood is backed-off [Katz, 1987] with the scaled unigram probability of the history-word, as follows:

$$P_{\text{TO}}^{\hat{}}(h = O(w_{i-k})|t = w_i) = \delta_b P(t = w_{i-k}) \quad (3.16)$$

This step implements a back-off mechanism, which should be scaled by a positive factor δ_b , such that the total probability mass is maintained to one. Notice, however, that the weight of the unigram probability can be arbitrarily assigned, as the likelihood will be properly normalized when computing the posterior probability. Alternatively, Laplace smoothing can be used to add a small additive count to all word-pairs. Hence the TO likelihood can be computed as:

$$P_{\text{TO}}^{\hat{}}(h = O(w_{i-k})|t = w_i) = \frac{\#(h = O(w_{i-k}), t = w_i) + \delta_a}{\#(t = w_i) + |V|\delta_a} \quad (3.17)$$

where δ_a denotes the additive count, typically $\delta_a = 1$.

3.3 Summary

This chapter has presented an approach to exploit long context for language modeling based on the distance and the occurrence events of word-pairs. The proposed model, referred to as the term-distance term-occurrence (TDTO) model, exploits the distance and

the occurrence events of a given word-pair jointly, hence formulating a more principled approach to exploiting long context information. To tackle the data scarcity problem in the TDTO model, the TDTO model is decoupled into two components: TD and TO, while the TD component exploits more specific distance events, the TO component exploit less specific events aiming at compensating the complexity in estimating the probability function.

Chapter 4

Evaluation and Analysis of the TDTO model

In Chapter 3, the term-distance term-occurrence (TDTO) model has been introduced. The TDTO model is formulated to exploit the events of term-distance (TD) and term-occurrence (TO), which correspond to the distances and occurrences of words in the context. By incorporating the inter-dependencies between the events, the TDTO model attempts to utilize information captured from the long context more optimally.

In this chapter, the evaluation of the TDTO model is discussed, particularly, the improvement gained by the model from exploiting longer context is measured. In Section 4.1, the perplexities of the TDTO model are evaluated in three experiments: (1) evaluation of the capability in exploiting long context, (2) comparison with the LLI combination of the n -gram and BOW models, and (3) comparison with the distant-bigram and trigger models. In Section 4.2, the applicability of the TDTO model is examined on three selected tasks: speech recognition, text classification and word prediction.

4.1 Perplexity Evaluation

This section discusses the experiment setup, corpora and results of the perplexity evaluation of the TDTO model. Specifically, experiments were conducted to assess the capability of the model of exploiting the long context and the comparison with other types of language models. The experiments were organized as follows:

1. **Experiment 1: TDTO for modeling the long context:** In this experiment, the perplexities of the TDTO model were measured for different lengths of contexts to evaluate its capability in exploiting the long context. As the TDTO model contains a trigram component, the improvement gained by the model can be observed from the reduction of perplexity.
2. **Experiment 2: TDTO vs. LLI of n -gram and BOW:** In this experiment, the TDTO model was compared with the LLI of the n -gram and BOW models. The intention was to examine if the TDTO model, which is formulated by exploiting the joint event of sub-sequence of words and BOW, performed better than the LLI model, which assumes both phenomena to be independent. The experiment results showed that the TDTO model performed better than the conventional approach.
3. **Experiment 3: TDTO vs. distant-bigram and trigger:** In this experiment, the TDTO model was compared with two other models that also exploit the long context, i.e. the distant-bigram and trigger models. The results again showed that the TDTO model outperformed the two competing techniques.

4.1.1 Corpora

The experiments were conducted on two different corpora to evaluate the performance of the TDTO model on languages in different styles:

1. **BLLIP Wall Street Journal [Charniak et al., 2000]:** This corpus consists of sentences collected from the Wall Street Journal (WSJ) from 1987 to 1989. The

data is partitioned as follows.

Table 4.1: Partitioning of the WSJ corpus

	train-set	development-set	test-set
#sentences	740k	1.2k	1.2k
#words	17M	30k	30k

2. **Switchboard-1** [Godfrey and Holliman, 1997]: This corpus consists of telephone conversation transcripts. The data is partitioned as follows.

Table 4.2: Partitioning of the SWB corpus

	train-set	development-set	test-set
#sentences	309k	2.7k	2.6k
#words	3.4M	35k	35k

For each corpus, a vocabulary was built from words selected randomly in the data. Rather than including all words into the vocabulary, the intention was to keep the complexity in the models manageable. The resulted vocabularies for the WSJ and SWB corpora contained 18k and 4k words, respectively, which caused about 5% out-of-vocabulary (OOV) rate in the data. Note that encountering OOV words is unavoidable in many applications, especially the speech recognition systems. It is a common practice in language modeling to substitute an OOV word by an unknown token, i.e. <UNK>, thus for perplexity evaluation, probabilities are estimated based on context which might contains <UNK> tokens.

4.1.2 Model Settings

In the experiments, the TDTO model was built based on the formulation depicted in Equation (3.10):

$$\begin{aligned}
P_{\text{TDTO}}(t = w_i | h_{\text{TDTO}} = w_{i-n+1}^{i-1}) \\
= \frac{\left(\begin{array}{c} P_{3\text{G}}(t = w_i | h_{3\text{G}} = w_{i-2}^{i-1})^{\beta_{3\text{G}}} \\ \prod_{k=1}^{n_{\text{TD}}-1} P_{\text{TD}-k}(h_{\text{TD}-k} = \Delta(w_{i-k}) | h_{\text{TO}-k} = \text{O}(w_{i-k}), t = w_i)^{\beta_{\text{TD}}} \\ \prod_{k=1}^{n_{\text{TO}}-1} P_{\text{TO}-k}(h_{\text{TO}-k} = \text{O}(w_{i-k}) | t = w_i)^{\beta_{\text{TO}}} \end{array} \right)}{Z(h_{3\text{G}}, h_{\text{TD}-1}, h_{\text{TD}-2}, \dots, h_{\text{TO}-1}, h_{\text{TO}-2}, \dots, \beta_{3\text{G}}, \beta_{\text{TD}}, \beta_{\text{TO}})} \quad (4.1)
\end{aligned}$$

where t denotes an arbitrary target-word w which the probability to be estimated by the TDTO model, i.e. $P_{\text{TDTO}}(\cdot|\cdot)$, based on the context h_{TDTO} . Each model component is configured as follows:

- The trigram model, $P_{3\text{G}}(\cdot|\cdot)$, considers the context $h_{3\text{G}}$ of length two. The model was trained by using the modified Kneser-Ney smoothing.
- The term-distance (TD) likelihoods, $P_{\text{TD}-k}(\cdot|\cdot, \cdot)$ for $k = 1, 2, \dots, n_{\text{TD}} - 1$, consider the context $h_{\text{TD}} = \bigcup_{k=1}^{n_{\text{TD}}-1} h_{\text{TD}-k}$ of length $|h_{\text{TD}}| = n_{\text{TD}} - 1$. For the smoothing, the moving average filter shown in Equation (3.12) was applied, with $\alpha = 0.1$. If the TD events of all distances, i.e. $k = 1, 2, \dots, n_{\text{TD}} - 1$ are unseen, it's assumed there existed one instance of the TD event in the distance of n_{TD} . Hence, following Equation (3.14), the smoothed count the considered TD event was $0.1^{n_{\text{TD}}-k}$.
- The term-occurrence (TO) likelihoods, $P_{\text{TO}-k}(\cdot|\cdot)$ for $k = 1, 2, \dots, n_{\text{TO}} - 1$, consider the context $h_{\text{TO}} = \bigcup_{k=1}^{n_{\text{TO}}-1} h_{\text{TO}-k}$ of length $|h_{\text{TO}}| = n_{\text{TO}} - 1$. For the smoothing, the likelihood of an unseen TO event was estimated from the unigram probability, see Equation (3.16). The back-off weight was fixed as $\delta_b = 0.1$.

The interpolation weights: $\beta_{3\text{G}}$, β_{TO} and β_{TD} were optimized by using a gradient descent algorithm.

4.1.3 Experiment 1: TDTO for modeling long context

This experiment examined the capability of the TDTO model in exploiting the long context. The perplexities of the TDTO model were measured for various lengths of context, i.e. $|h_{TD}| = |h_{TO}| = 0, 1, 2, \dots, 10$. The perplexity results are shown in Figure 4.1.

Figure 4.1: On the WSJ and SWB corpora, the perplexities of the TDTO model were reduced up to 11.2% and 6.5%, respectively, when longer context was considered. Also, the reduction of perplexities within the trigram context indicates that the TD and TO components are capable of capturing complementary information from the trigram context.

Overall, the perplexities of the TDTO model were consistently reduced as longer TD and TO’s contexts were considered, i.e. from one to five. Specifically, on the WSJ corpus, the perplexity was reduced up to 11.2%, from 130.1 to 115.5, while on the SWB corpus, 6.5%, from 81.7 to 76.3. These reductions were obtained with the context lengths of eight and seven, respectively.

However, when farther context was included, no significant reduction was observed. For the case of SWB, exploiting the context of lengths from seven to ten deteriorated the model performance. This is explained by the average size of sentences in the SWB corpus, which is 13 words; as compared to the WSJ corpus in which the average sentence size is 25 words. Moreover, up to 53% of the sentences in the SWB corpus are shorter

than 7 words, while in the WSJ corpus, only 5%. That is to say, for many sentences in the SWB corpus, the considered context is larger than the sentence itself. Under this situation, the considered context stopped at the sentence boundary and the far context was unavailable to give any improvement.

Also, the results shown in Figure 4.1 suggest that not only the TD and TO information derived beyond the trigram context, i.e. $|h_{TD}| > 2$ and $|h_{TO}| > 2$, improved the model performance. The TD and TO information that was captured from the same or shorter context than the trigram, i.e. $|h_{TD}| \leq 2$ and $|h_{TO}| \leq 2$, improved the overall TDTO model performance too (see the perplexities plots at the left-hand-side of $|h_{TD}| = |h_{TO}| = 2$). This highlights the advantage of the TDTO model, which is able to capture complementary information within the trigram context that has not been properly captured by the trigram model itself. This matter will be further discussed.

Interpolation weights to the context lengths

In addition to the perplexity evaluation results, the behavior of the interpolation weights: β_{3G} , β_{TD} and β_{TO} , was examined. As the weights were optimized to minimize the perplexity of the model (as shown in Figure 4.1), this analysis gives an insight of how the trigram, TD and TO model components interact, particularly, to give the minimize perplexity. The weight configurations are shown in Figure 4.2.

Figure 4.2: Optimal weights for the trigram (3G), TD, and TO model components as corresponding to the perplexities depicted in Figure 4.1, for the WSJ and SWB corpora.

From the plots of the TD and TO’s weights, it is clear how the TD and TO model components’ relative importance changed with respect to the context length. The analysis suggests that probabilities can be more accurately estimated by putting different emphasis on the distance and occurrence attributes, which is possible due to the proposed decoupling operation (see Section 3.2.1).

As shown in Figure 4.2, the weights of the TD and TO components first increased for a short context and then gradually decreased as the context length grew. As far context is more likely to contain TD and TO events that are weakly associated to the target-words, the weights have to be lowered to lessen the impact. The smoothing methods, which are based on moving average filtering for the TD, see Equation (3.12), and a backing-off mechanism for the TO, see Equation (3.16), are inadequate in handling the noisy TD and TO events at the far context. This explains why the reduction of perplexities (as shown in Figure 4.1) became insignificant when the considered context was expanded beyond certain length, e.g. $|h_{TD}|, |h_{TO}| \geq 5$. From the plots of the TO’s weight, the TO information in the near context, e.g. $|h_{TO}| < 5$, is seen to be more useful to the SWB text. Higher amount of TO information was needed to compensate for the relatively weaker SWB’s trigram component.

Moreover, the comparatively higher weights as required by the trigram components shows the importance of the inter-dependencies among the words in the context, which has been relaxed in the TD and TO model components (see Section 3.2.2). Also, notice that the trends of the trigram’s weights were in the opposite directions from the TD and TO’s weights as the context expanded. This observation shows that the trigram information became relatively more important to compensate the TD and TO components that became less reliable as the context grew longer, i.e. $|h_{TD}|, |h_{TO}| > 5$.

TD and TO for alleviating scarcity problem

As seen from Figure 4.1, the TDTO model showed a lower perplexity even when the context considered by the TD and TO components was as short as one or two. In

such cases, the TD and TO considered context of the same or shorter length than the trigram, i.e. $|h_{TD}|, |h_{TO}| \leq 3$. As the TDTO model contains a trigram component, see Equation (4.1), the observed improvements suggest that the TDTO model has captured complementary information that was not accounted for by the trigram model.

To better understand this scenario, consider the situation in which an unseen or rare context is encountered. In the traditional n -gram model scenario, the n -gram model is reduced to a lower order n -gram model, i.e. back-off mechanism, and the context across is discarded. The TDTO model overcomes this drawback by relaxing the dependencies among words in the context so that the far context is accessible without being back-off. Also, decoupling allows the contribution of the TD and TO to be adjustable as according to the language style. In Figure 4.1, the TDTO model of context length of two showed lower perplexity than the trigram model by 2.8% on the WSJ corpus while 4.4% on the SWB corpus. Higher gain as obtained on the SWB corpus implies a more severe data scarcity problem in training an n -gram model from the conversational transcripts which allows more room of improvement.

This issue was further investigated by fixing the length of the context considered by the three model components to be the same, i.e. $|h_{TD}| = |h_{TO}| = |h_{NG}|$. The TDTO model was then compared with an n -gram model that shared the same context, i.e. both models estimate probabilities based on the same context. The resulting perplexities on the WSJ and SWB corpora are shown in Figure 4.3.

As observed from Figure 4.3, the TDTO model consistently performed better than the n -gram model in terms of perplexity for all considered context lengths. There was a clear trend for the TDTO model when the context length was increased, larger gain of perplexity reduction was obtained. More specifically, on the WSJ corpus, the bigram’s perplexity was only reduced by 0.5%, from 197.9 to 197.0, whereas the sixgram’s perplexity was reduced up to by 9.0%, from 118.0 to 107.4. Similar trend was observed on the SWB corpus, where 1.4% reduction on the bigram’s perplexity, from 99.4 to 98.0, but 6.1% on the sixgram’s perplexity, from 79.7 to 74.8. Such phenomenon was due to a

Figure 4.3: The perplexities of the n -gram models (NG) of orders from two to six, i.e. context lengths from one to five, as compared with the TDTO models that share the same context. The results imply that the TDTO model component is capable of capturing complementary information in the n -gram context that the n -gram model alone is not able to capture.

more severe scarcity problem when modeling higher order n -grams.

4.1.4 Experiment 2: TDTO vs. LLI of n -gram and BOW

In this experiment, the TDTO model was compared with the LLI of the n -gram and BOW models. Although both models estimate probabilities similarly based on the subsequence of words and BOW, the TDTO model is formulated to exploit these events jointly, as an attempt to capture the inter-dependencies (as discussed in Section 3.2.1). The LLI model, on the other hand, assumes both events to be independent. This aim of this experiment was to examine whether by incorporating the inter-dependencies would improve the performance of the language model, particularly when exploiting the long context.

The LLI of the n -gram and BOW models can be formulated as follows.

$$\begin{aligned}
P_{3G-BOW}^{LLI}(t = w_i | h = w_{i-n+1}^{i-1}) \\
= \frac{P_{3G}(t = w_i | h_{3G} = w_{i-2}^{i-1})^{\beta_{3G}} P_{BOW}(t = w_i | h_{BOW} = \{w\}_{i-n_{BOW}+1}^{i-1})^{\beta_{BOW}}}{Z(h_{3G}, h_{BOW})}
\end{aligned}$$

where $P_{3G}(\cdot|\cdot)$ denotes a trigram model, while $P_{BOW}(\cdot|\cdot)$ denotes the BOW model. The interpolation weights for the n -gram and BOW models are denoted by β_{3G} and β_{BOW} , respectively. The BOW model was implemented following the latent semantic indexing (LSI) paradigm [Bellegarda, 1998, Coccaro, 2005, Novak and Mammone, 2001], hence is referred to as the LSI model.

Figure 4.4 shows the perplexities of the TDTO and LSI models of context lengths from one to ten, i.e. $|h_{TD}|, |h_{TO}|, |h_{BOW}| = 0, 1, 2, \dots, 10$. On both corpora, the TDTO model consistently showed lower perplexities in all considered context lengths. On the WSJ corpus, the perplexity of the TDTO model was reduced up to 11.2%, from 130.1 to 115.5, whereas the perplexity of the LSI model was reduced only by 3.9%, from 130.1 to 125.0. On the SWB corpus, the perplexity of the TDTO model was reduced up to 6.5%, from 81.7 to 76.3, while the LSI model showed only marginal reduction, i.e. 2.0%, from 81.7 to 80.1.

Figure 4.4: Resulting perplexities for the TDTO and LSI models of context lengths from zero to ten. On both corpora, the TDTO model showed lower perplexities as compared to the LSI model.

Lower perplexities shown by the TDTO model indicates the importance of incorporating the inter-dependencies into the language model, especially when exploiting the long context. Nevertheless, the strength of the LSI model lies on its capability in modeling documents, where each context might comprise up to hundred words. For language modeling, as the main intention is to facilitate the speech recognition systems, which

each utterance consists of only about 10-20 words, the advantage of the LSI model is undetermined. For validation, the LSI model was evaluated on the Gigaword corpus, which comprises newswire documents, the context can be extended up to 300 words. The tri-gram perplexity was reduced up to 17.5% which is consistent to the results reported in the original papers [Bellegarda, 1998, Coccaro, 2005, Novak and Mammone, 2001].

4.1.5 Experiment 3: TDTO vs. Distant-Bigram and Trigger

In this experiment, the TDTO model was compared with the distant-bigram (DBG) and the trigger (TGR) models to examine the advantages of exploiting long context by using the TD and the TO components.

The distant-bigram model

The perplexity results depicted in Figure 4.2 suggest that the TD and TO components possess different predictive powers towards a target-word. The distant bigram (DBG) model [Lau et al., 1993b, Huang et al., 1993, Simons et al., 1997], on the other hand, can be seen as a specific case of the TDTO model when the TD and TO components are assigned a common weight, i.e. $\beta_{\text{DBG}} = \beta_{\text{TD}} = \beta_{\text{TO}}$, and exploit a common context in the history, i.e. $h_{\text{DBG}} = h_{\text{TD}} = h_{\text{TO}}$, of length $n_{\text{DBG}} = n_{\text{TD}} = n_{\text{TO}}$. Follow the TDTO formulation in Equation (3.10), the TDTO model is reduced to a DBG model:

$$\begin{aligned}
& \frac{\left(\begin{array}{c} P_{\text{NG}}(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1})^{\beta_{\text{NG}}} \\ \prod_{k=1}^{n_{\text{DBG}}-1} P_{\text{TD}-k} \left(h_{\text{TD}-k} = \Delta(w_{i-k}) | h_{\text{TO}-k} = \text{O}(w_{i-k}), t = w_i \right)^{\beta_{\text{DBG}}} \\ \prod_{k=1}^{n_{\text{DBG}}-1} P_{\text{TO}-k} \left(h_{\text{TO}-k} = \text{O}(w_{i-k}) | t = w_i \right)^{\beta_{\text{DBG}}} \end{array} \right)}{Z(h_{\text{NG}}, h_{\text{TD}-1}, h_{\text{TD}-2}, \dots, h_{\text{TO}-1}, h_{\text{TO}-2}, \dots, \beta_{\text{NG}}, \beta_{\text{DBG}})} \\
& = \frac{\left(\begin{array}{c} P_{\text{NG}}(t = w_i | h_{\text{NG}} = w_{i-n_{\text{NG}}+1}^{i-1})^{\beta_{\text{NG}}} \\ \prod_{k=1}^{n_{\text{DBG}}-1} P_{\text{DBG}-k} \left(h_{\text{DBG}-k} = \Delta(w_{i-k}) | t = w_i \right)^{\beta_{\text{DBG}}} \end{array} \right)}{Z(h_{\text{NG}}, h_{\text{DBG}-1}, h_{\text{DBG}-2}, \dots, \beta_{\text{NG}}, \beta_{\text{DBG}})} \quad (4.2)
\end{aligned}$$

where both types of information, i.e. TD and TO, are treated equally important and the coupled events are used directly for modeling each bigram probability. External knowledge about distance is coarsely introduced into the model through a weighting scheme, which is performed during the model interpolation. The formulation of a log-linear interpolated DBG model is as follows.

$$\begin{aligned}
P_{\text{DBG}}(t = w_i | h_{\text{DBG}} = w_{i-n_{\text{DBG}}+1}^{i-1}) \\
= \frac{P_{1\text{G}}(t = w_i)^{\beta_{1\text{G}}} \prod_{k=1}^{n_{\text{DBG}}-1} P_{\text{DBG}-k}(t = w_i | h_{\text{DBG}-k} = w_{i-k}^{\beta_{\text{DBG}-k}})}{Z(h_{\text{DBG}-1}, h_{\text{DBG}-2}, \dots, \beta_{\text{DBG}-1}, \beta_{\text{DBG}-2}, \dots)} \quad (4.3)
\end{aligned}$$

where n_{DBG} is the context length and $\beta_{\text{DBG}-k}$, for $k = 1, 2, \dots, n_{\text{DBG}}-1$, is the weight assigned to the distance- k bigram. Note that the unigram probability $P_{1\text{G}}(t = w_i)$ is introduced to compensate the overestimation of the target-word probability, as incurred by each distance- k bigram model, which implicitly contains a prior. In general, the weight of the unigram model $\beta_{1\text{G}}$ holds a negative value. Such compensation of the prior is not required in the TDTO model, as the posterior is computed as a product of likelihood scores and the prior is only introduced once by the n -gram model, see Equation (3.10).

When interpolated with a trigram model, the DBG model is formulated as follows:

$$\begin{aligned}
P_{3\text{G-DBG}}(t = w_i | h = w_{i-n+1}^{i-1}) \\
= \frac{P_{3\text{G}}(t = w_i | h_{3\text{G}} = w_{i-2}^{i-1})^{\beta_{3\text{G}}} \prod_{k=1}^{n_{\text{DBG}}-1} P_{\text{DBG}-k}(t = w_i | h_{\text{DBG}-k} = w_{i-k}^{\beta_{\text{DBG}-k}})}{Z(h_{3\text{G}}, h_{\text{DBG}-1}, h_{\text{DBG}-2}, \dots, \beta_{3\text{G}}, \beta_{\text{DBG}-1}, \beta_{\text{DBG}-2}, \dots)} \quad (4.4)
\end{aligned}$$

where $\beta_{3\text{G}}$ is the interpolation weight of the trigram model. The overall context length is $n = \max(2, n_{\text{DBG}})$, typically $n = n_{\text{DBG}}$.

The trigger model

In addition, the TDTO model was also compared to the trigger (TGR) model which exploits well associated word-pairs for language modeling. As mentioned earlier, as the trigger model considers only the words' occurrences, it is similar to the TO model. How-

ever, it filters out the weak associated word-pairs and multiple occurrences of the same word in the history-context are counted only once.

Traditionally, statistics of the trigger-pairs are incorporated into language model through the maximum entropy framework [Lau et al., 1993b, Rosenfeld, 1996]. In this experiment, however, a log-linear interpolation with a trigram model is implemented to compare fairly with the TDTO model.

$$P_{3G-TGR}(t = w_i | h_{TGR} = w_{i-n_{TGR}+1}^{i-1}) \\ = \frac{P_{3G}(t = w_i | h_{3G} = w_{i-2}^{i-1})^{\beta_{3G}} \prod_{k=1}^{n_{TGR}-1} P_{TGR-k}(t = w_i | w_{i-k} \in h_{TGR})^{\beta_{TGR}}}{Z(h_{3G}, h_{TGR}, \beta_{3G}, \beta_{TGR})} \quad (4.5)$$

where n_{TGR} is the context length and β_{TGR} is the weight assigned to the trigger model. The trigger-pairs are pre-selected from the training set as word-pairs that possess positive point-wise mutual information.

Evaluation on exploiting distant context

For varying sizes of context of history, the performance of the TDTO model, see Equation (4.1), is compared with the DBG model and the TGR model, as depicted by Equation (4.4) and (4.5), respectively. Evaluated on the test-sets of the WSJ and SWB corpora, Figure 4.5 shows the perplexities of these models, of history lengths from zero to ten (history length of zero indicates a plain trigram model). Similar to Figure 4.1, perplexities on the WSJ and SWB corpus are plotted separately on the left and right vertical axes, and the horizontal axis denotes the history length.

On the WSJ corpus, the TDTO and DBG models showed comparable results, i.e. outperformed trigram perplexities up to 11.2% and 9.0%, respectively. However, the TGR model showed a milder reduction as compared to the TDTO and DBG models, i.e. 6.4%. On the SWB corpus, the DBG model could only slightly reduce the trigram perplexity by 1.1% (from 81.7 to 80.7), while the TDTO model has shown a reduction of up to 6.5% (from 81.7 to 76.3). The TGR model showed a much comparable result to the TDTO model, for which the perplexity was reduced up to 4.1%. It is also interesting

Figure 4.5: Resulting perplexities for TDTO and DBG models of history lengths from zero to ten. On the WSJ corpus, both results are comparable. On the SWB corpus, the TDTO model reduced the perplexity up to 6.5% while the DBG model showed only marginal improvement.

to notice that the DBG model outperformed the TGR model on the WSJ corpus but a contrary result was seen on the SWB corpus. As the SWB corpus contains transcriptions of conversational speech, where word regularities are more difficult to be captured, by ignoring the word position and weak associated word-pairs, the trigger model has an advantage over the DBG model that exploits the word-pairs in given fixed distances.

In addition, it can also be seen from Figure 4.5 that word-pairs with distances greater than two do not contribute to perplexity reduction in the case of DBG modeling. However, when distance and co-occurrence information are decoupled and properly weighted, as in the case of the TDTO model, farther history is more effectively exploited by the model.

Improving higher-order n -gram model

Next, experiment was conducted to evaluate contribution of the TDTO and DBG models to alleviating the scarcity problem of the n -gram model. Similar to previous experiments (see Figure 4.1), perplexities for TDTO, DBG and TGR models when both the baseline n -gram model and the complementary model are looking at the same history-context. In the conducted experiments, history-context lengths were varied from 2 to 5. The results

are shown in Figure 4.6.

Figure 4.6: On the WSJ corpus, perplexity reductions to the n -gram models (NG), when interpolated with TDTO, DBG and TGR models looking at the same history-context lengths. On the SWB, TDTO model outperformed both DBG and TGR models.

As seen from the figure, on the WSJ corpus, all models performed comparably, and the TDTO model did not show significant advantage over the DBG model. This can be due to two reasons: the large amount of training data available for the WSJ corpus and the well grammatically formed texts composing this dataset. On the other hand, on the SWB corpus, the TDTO model consistently outperformed both, the DBG and the TGR, models throughout all considered history lengths.

From the implementation point of view, the TDTO model exhibits an additional advantage over the DBG. When interpolating the TDTO model with the n -gram model, there are only three weights that are required to be adapted: β_{NG} , β_{TD} , and β_{TO} . On the other hand, for the DBG model, $n_{\text{DBG}} + 1$ interpolation weights have to be tuned: β_{NG} , $\beta_{1\text{G}}$, and $\beta_{\text{DBG}-k}$ for $k = 1, 2, \dots, n_{\text{DBG}} - 1$. Grid searching with fine resolution is memory-wise and computationally expensive. A much more complex and expensive optimization procedure is then required for tuning the DBG model parameters.

In the case of the SWB corpus, less training material is available, and it consists of text that is less grammatically formed and more spontaneously produced. Under such restricted data scenario, using the coupled word-pair directly for prediction (as in the DBG modeling) had been seen to improve the baseline n -gram performance insignifi-

cantly. However, by decoupling the distance and co-occurrence information in word-pairs, the TDTO model has shown to provide a much better performance improvement with respect to the baseline n -gram model.

4.2 Applications of the TDTO Model

In Section 4.1, the proposed TDTO model has been empirically shown to significantly outperformed the n -gram model in terms of perplexity. Also, the TDTO model has been shown to exploit long context more effectively as compared to the LSI, distant-bigram and trigger models. However, although the perplexity is a good measure of model quality, it does not necessarily relate to model’s performance on a given task. For this reason, the performance of the TDTO model was also evaluated by using task-dependent metrics. In this section, how the proposed model implementation impacts the performance of some applications is evaluated. In particular, experiments were conducted on three applications: speech recognition, document classification, and word prediction.

4.2.1 Application 1: Speech Recognition

In this subsection, the performance of the proposed TDTO model is evaluated in improving the word error rate (WER) of speech recognition by means of an N -best re-ranking task. More specifically, for each test utterance, an N -best list of hypotheses is first generated by using the HTK’s HVite decoder [Young et al., 2006] and then, the language model scores are recomputed by using the TDTO model. Hypotheses with the best score were selected as the recognized utterances. The WER obtained with an n -gram model served as baseline to be compared with the TDTO interpolated model result.

The experiment was conducted on the Aurora-4 corpus [Parihar et al., 2015]. The training set was composed of 7,138 utterances and 330 utterances were used as cross-validation data. A two-fold cross validation was conducted with the 330-utterance set: one subset was used for fine-tuning the interpolation weights: β_{NG} , β_{TD} , and β_{TO} , see

Equation (4.1), while the other subset was used for evaluation, and vice-versa in the second fold. The training set was used to train the acoustic model, which is based on 39-dimension MFCCs feature built on 3-state triphone HMMs. Both language models: the baseline n -gram and TDTO were trained with a separate dataset apart from the transcriptions of the Aurora-4 corpus. However, to keep the domain consistent, the BLLIP WSJ corpus was used for the training. The vocabulary was the standard 5K word list used in the Aurora-4 task. In order to observe the gain contributed by the TDTO model with respect to the n -gram model’s performance, the WER obtained by using the n -gram models alone was first computed as baselines. The orders were considered from two to six. Then, each of the n -gram models were combined with a TDTO model considering a history-context length of five. The choice of the history length was based on the previous observation (see Figure 4.1), where the reduction of perplexity started to saturate. Nevertheless, the optimum length may be longer (as discussed in Section 4.1.3). The results are presented in Figure 4.7.

As shown in Figure 4.7, the TDTO model consistently improved the WER obtained by n -gram models of various orders. As the TD and TO information were captured from a history-context of length five, for lower order n -gram model, the WER was improved mainly by the TD and TO information located beyond the n -gram context. For higher order n -gram model such as the six-gram model, as the history-context considered by both models was the same, the WER was improved by the complementary information captured by the TD and TO components within the same n -gram context. These WER improvements are consistent with perplexity reductions observed in Section 4.1.3.

Also, the experiment is repeated by replacing the TD and TO components with a distant bigram model, see Equation (4.4), that also considered a history context of length five. However, except when combined with the bigram model for which the WER was reduced 9.7% relatively, no improvement was observed for other higher n -gram orders. This suggests, as in Section 4.1.5, that the distant bigram model is not as effective as the TDTO model in providing the n -gram model with complementary information when

Figure 4.7: Word error rate (WER) resulting from n -gram models (NG) of order from two to six, and the respective interpolated TDIO models. The TDIO models consistently improved the WER obtained by using the n -gram alone, with relative improvements (RI) of 4.6-9.7%.

given the same context. Again, this result shows the value of the decoupling operation.

4.2.2 Application 2: Text Classification

This subsection presents an experiment carried out to assess whether the TDIO model is also useful for text classification. The idea was motivated by previous works of using n -gram models to discriminate text into categories of genre [Fürnkranz, 1998, Steffen, 2004, Bermingham and Smeaton, 2010], authorship [Peng and Schuurmans, 2003], sentiment [Pang et al., 2002, Bermingham and Smeaton, 2010], language [Vatanen et al., 2010], etc. Such discriminative capability is believed can be also exhibited by the TD and TO model components. For example, text prepared in a template-based format, such as stock market reports or weather forecasts, exhibit some degree of consistency on word relative

positions. To distinguish such text from others, the distances between words, as modeled by TD, would serve as an effective feature.

The experiments presented here were conducted on the ModApte split of the Reuters-21578 dataset¹ and the Cornell University’s polarity dataset v2.0 [Pang and Lee, 2004]. For Reuters-21578, the ten categories with the most data available are selected. The vocabulary consisted of the 13K most frequent words in the dataset. For polarity, training sets and test sets for both, positive and negative, categories consisted of 900 and 100 randomly selected documents, respectively.

Experiments were conducted as binary classification tasks. Indeed, a test document D is classified as belonging to a given category C^+ , against the non-category C^- , based on the likelihood ratio:

$$c^* = \begin{cases} C^+ & \text{if } \frac{P(D|C^+)}{P(D|C^-)} > k_{\text{th}} \\ C^- & \text{otherwise} \end{cases} \quad (4.6)$$

where the threshold k_{th} trades off the false alarm and miss rates, and is to be tuned empirically.

The likelihood score of a document is computed as the product of probabilities of all words in the document:

$$P(D|c) = \prod_{w \in D} P_c(t = w|h(w)) \quad (4.7)$$

where the term P_c refers to the language model trained on data belonging to the class C .

The performance of different models are compared on the two proposed classification tasks, including the unigram, the six-gram, the 1G-TD, and the 1G-TO, where 1G-TD and 1G-TO correspond to the TD and the TO component models interpolated with an unigram model. For the TO and TD model components, the history length was set to 5. The interpolation weights of the TD and TO model components were both set to $\beta_{\text{TD}} = 0.1$ and $\beta_{\text{TO}} = 0.1$ while the weight of the unigram model component was set to $\beta_{\text{1G}} = 0.9$. The choices of the history lengths and the interpolation weights of the TD and TO model components were selected ad-hoc, hence no fine-tuning was conducted. The

¹<http://www.daviddlewis.com/resources/testcollections/reuters21578/>

Table 4.3: F1 score for each category in the Reuter-21578 dataset as obtained by the unigram, sixgram, 1G-TD, and 1G-TO models.

Category	1G	6G	1G-TD	1G-TO
ACQ	0.961	0.957	0.963	0.840
CORN	0.459	0.424	0.473	0.297
CRUDE	0.899	0.764	0.907	0.513
EARN	0.962	0.974	0.969	0.940
GRAIN	0.849	0.833	0.856	0.642
INTEREST	0.661	0.629	0.708	0.375
MONEY-FX	0.821	0.807	0.821	0.498
SHIP	0.580	0.398	0.580	0.204
TRADE	0.618	0.634	0.667	0.419
WHEAT	0.548	0.470	0.546	0.294
MicroAve	0.853	0.820	0.864	0.635
MacroAve	0.736	0.689	0.749	0.749

motivation here is merely to look into the potential of the TD and TO model components for text classification.

The resulting F1 scores for each category with the different models, and their corresponding overall average scores are shown in Table 4.3 and Table 4.4, for both the Reuters and polarity tasks, respectively.

From the results of the unigram and the six-gram models, longer context, five pre-

Table 4.4: F1 score in the polarity dataset as obtained by the unigram, sixgram, 1G-TD, and 1G-TO models.

Category	1G	6G	1G-TD	1G-TO
Average	0.814	0.849	0.849	0.774

ceding words in this case, performs contradictorily in both tasks of text classification. Longer context, which includes the distance and co-occurrence information, deteriorated the genre classification task (as seen in Table 4.3), but improved the performance of the polarity classification task (as seen in Table 4.4). Nevertheless, the TD information consistently improved the performance of the unigram model in both tasks. Particularly, in the Reuters task (see Table 4.3), besides improving the performance in most classes, the scores in INTEREST and TRADE classes have been raised up to 7% and 8% relatively. Such results show that the word-pair distance information provided by the TD model can be used as a more discriminating attribute than the n -gram model, which combines both distance and co-occurrences but in a more scarce sensitive way.

Although the TO model component did not perform well in both classification tasks, several works have stressed the significance of using word association in text classification tasks [Antonie and Zaiane, 2002, Cao et al., 2005, Wu and Wang, 2012]. These works utilized various methods, including human annotations and some similarity measures, to exploit word association information in the classification task. In this case, as the TO model component was derived directly from the raw counts, it also included non-associated words as seen in the data.

To further improve these results, the three components: n -gram, TD and TO can be combined in order to underline the use of different information for classification. The interpolation weights, however, need to be optimized discriminatively for stretching the inter-class differences while maintaining the intra-class robustness. Moreover, more powerful classifiers can be attempted, such as support vector machine and neural networks, which have shown to achieve F1 score of over 0.9 in both datasets [Dumais et al., 1998, Kim, 2014].

4.2.3 Application 3: Word prediction

The perplexity reduction results presented previously showed the advantage of the TDTO model with respect to the n -gram model. In this last experiment, a word predic-

tion task is evaluated, in which the subsequent word must be predicted given the preceding context. This task is closely similar to other general word prediction tasks [Wandmacher and Antoine, 2008, Rosa, 2002] and some similar efforts for the purpose of spelling correction [Thiele et al., 2000, Duan and Hsu, 2011] and keystrokes savings [Matiasek and Baroni, 2003, Roark et al., 2010].

For this task, experiment was conducted to evaluate whether the TDTO model is able to improve the baseline n -gram model performance. Candidates were selected as the twenty most frequent words in the corpus. As each candidate word occurs many times within the training data, a large variety of history-contexts are actually encountered. Hence, a good performance on the prediction task reflects the actual discriminative capability possessed by the TDTO model for the specific case of function words. This task, which is similar to a previous work on recovering missing frequent words in text translations [Wandmacher and Antoine, 2008], has a huge impact as it affects a large proportion of the running words in a given corpus.

The experiment was conducted on the BLLIP WSJ corpus [Charniak et al., 2000]. Similar to previous experiments, the same data splitting scheme was used: the training set was used to build the n -gram and TDTO models, while the test set was used to compute the prediction accuracy, and the interpolation weights were optimized by using the development set. Different from the setting in previous experiments, where the weights were adjusted to fit the models to the text, i.e. to yield the lowest perplexity, the weights were tuned to improve the performance of the task under consideration.

To predict the subsequent word, the candidate target-word that maximizes its probability given the observed history-context was selected:

$$w^* = \operatorname{argmax}_{w_i \in V_c} P(t = w_i | h = w_{i-1}^{i-n+1}) \quad (4.8)$$

where V_c denotes the set of candidate words. $|V_c| = 20$ in this experiment. The n -gram model was used as the baseline to be compared with the TDTO interpolated model. As the aim was to find the most likely word by maximizing its conditional probability, 4.1 can be simplified as follows:

$$w^* = \underset{w_i \in V_c}{\operatorname{argmax}} \left(\begin{array}{c} P_{3G}(t = w_i | h_{3G} = w_{i-2}^{i-1}) \\ \prod_{k=1}^{n_{TD}-1} P_{TD-k}(h_{TD-k} = \Delta(w_{i-k}) | h_{TO-k} = O(w_{i-k}), t = w_i)^{\beta_{TD}} \\ \prod_{k=1}^{n_{TO}-1} P_{TO-k}(h_{TO-k} = O(w_{i-k}) | t = w_i)^{\beta_{TO}} \end{array} \right) \quad (4.9)$$

The order of the n -gram model is fixed to two, i.e. $n_{NG} = 1$, and the history length of the TDTO model to five, i.e. $n_T = 5$. Evaluated on the the BLLIP WSJ corpus, the accuracy of the bigram model was improved 7.0% relatively, from 47.1% to 50.4%. The weight of the n -gram component β_{NG} was fixed to one. The weights of the TD and TO component models had been fine-tuned to be $\beta_{TD} = 0.65$ and $\beta_{TO} = 0.4$, by using the development-set. Such setting shows the uneven amount of TD and TO information required for this word prediction task. The result highlights the benefit of the proposed TDTO formulation which allows for decoupling the TD and TO components.

4.3 Summary

This chapter has presented the results of evaluation of the TDTO model. The perplexity of the TDTO model has been evaluated on the WSJ and SWB corpora. Three key findings are discovered:

1. The TDTO model is capable to exploit the distant context to reduce the trigram model perplexity, up to 11.2% and 6.5%, on the WSJ and the SWB corpora. Empirical results show the TDTO model provides complementary information within the n -gram context which is not captured by the n -gram model.
2. As the TDTO model is formulated by exploiting the n -gram and the BOW jointly, comparison to the LLI model of n -gram and BOW has shown the TDTO possess lower perplexity. This finding highlights the importance of joint modeling of the representations of the history.

3. The TDTO model is compared to two other models which exploit also the distant context, i.e. the distant-bigram and the trigger models. The TDTO model has been shown to outperform these two models.

In addition to the perplexity reduction results, the TDTO model has been applied to the following three tasks: speech recognition, text classification, and word prediction. For each task, the TDTO model shows significant improvements when compared to a baseline model.

However, the TDTO model has a weakness in smoothing: the TD and the TO component models are smoothed by using a running average filter and back-off mechanism which are weak theoretical founded. The smoothing methods are applied based on intuition as an ad-hoc to avoid zero probability problem. Currently, the running average filter applied to the TD component has the deficiency of not exploiting the syntactic constraints exhibited by words in language. For example, some selected word-pairs, due to their POSs, may exhibit more consistent distances than others. On the other hand, the occurrence counts of the TO component are to be more appropriately smoothed to reflect the underlying word distribution, e.g. as according to a power distribution.

Chapter 5

The Neural Network Approach to TDTO Language Modeling

In Chapter 3, the term-distance term-occurrence (TDTO) model has been introduced. The TDTO model consists of two primary components: term-distance (TD) and term-occurrence (TO), that are used to compute likelihood scores for the distances and occurrences of words, respectively. For the smoothing, the TD component uses a running average filter to redistribute the observed counts based on the distances of word-pairs, while the TO component reduces an unseen word-pair to an unigram, i.e. the back-off mechanism.

In this chapter, a neural network implementation of the TDTO model is discussed. The intention is to address the smoothing problem with the TDTO modeling – it is arguable that the original smoothing techniques, i.e. the running average filter and the back-off mechanism, are relatively simple to describe the relations among words in the language. The advantage of using neural networks for the TDTO modeling lies in the representations of the TD and TO events in the continuous space, from which more accurate probabilities can be estimated. Particularly, the relations among the TD or TO events are learned by using a neural network such that “similar” events will be located closely in the continuous space.

Section 5.1 provides the background for neural networks language modeling. Section 6.2.2 presents the implementation of the TDTO model by using a feedforward neural network. The resulted model, coined as the NN-TDTO model, is then compared with the baseline TDTO model in terms of perplexity and word error rate (WER) on a speech recognition system. The results are presented in Section 5.3. The NN-TDTO model is also compared with the conventional feedforward neural network based language model. This chapter is summarized in Section 5.4.

5.1 Background

Smoothing is crucial to language modeling. Due to the data scarcity problem, there are many rare and unseen events, which the probabilities cannot be estimated accurately. The idea of smoothing is to adjust the distribution such that “similar” events would be given similar probabilities. For example, given the following two contexts are “similar”,

a collapse in the blue chip stocks
the falling of the blue chip stocks

probabilities estimated based on these contexts shall be given the similar values:

$$P(\textit{could} \mid \textit{a collapse in the blue chip stocks})$$

$$P(\textit{could} \mid \textit{the falling in the blue chip stocks})$$

Traditional smoothing methods attempt to redistribute the probability mass, such that the probability of a rare or unseen event is computed by using the statistics of other “similar” events that are more frequently seen. However, determining similarity among categorical events, such as context, is difficult. Typically, contexts that contain the same sub-sequence of words are assumed to be “similar”. For example, the probability of a rare or unseen n -gram is computed from the counts of the lower orders n -grams [Katz, 1987, Witten and Bell, 1991, Ney et al., 1994, Kneser and Ney, 1995] (see Section

2.2.1). The similarity, as assumed by these approaches, constitutes the local continuity in the categorical domain, which subsequently contributes to the smoothness in the probability function.

Neural networks address the smoothing problem with language modeling by representing the categorical events in the continuous space, where the similarity can be learned from the data. Specifically, as the representation vectors of the “similar” events would be closely located, a smooth probability function can be obtained without making any assumption on the context [Bengio et al., 2003, Schwenk, 2007]. In general, computing probabilities by using a neural network language model can be depicted as follows:

$$P(t = w_i | h = w_{i-n+1}^{i-1}) = P(t = w_i | \mathbf{h} = f_{\text{context}}(w_{i-n+1}^{i-1})) \quad (5.1)$$

where $f_{\text{context}}(\cdot)$ denotes a function that computes the representation vector $\mathbf{h}$ based on the context $h = w_{i-n+1}^{i-1}$. Typically, the dimension of the continuous space, i.e. $|\mathbf{h}|$, is fixed to be lower by orders of magnitude, e.g. 50, than the vocabulary size, e.g. 50k, so that the similarity can be exploited more effectively.

In general, there are two categories of neural network approaches to language modeling: feedforward neural network (FNN) and recurrent neural network (RNN). The main difference between them lies in how a context is mapped to a representation vector in the continuous space. These two categories of approaches are elaborated as follows:

1. Feedforward neural network (FNN):

A FNN-based language model feeds the words in the context in parallel into the network, thus the context-vector is computed as the concatenation or combination of the word-vectors. For modeling the n -grams, the word-vectors which correspond to the $(n - 1)$ words in the preceding context are concatenated [Bengio et al., 2003, Schwenk, 2007].

$$\mathbf{h} = \langle f_{\text{word}}(w_{i-1}), f_{\text{word}}(w_{i-2}), \dots, f_{\text{word}}(w_{i-n+1}) \rangle \quad (5.2)$$

where $f_{\text{word}}(\cdot)$ denotes the mapping function for words.

For modeling the BOW, the word-vectors are linearly combined [Mikolov, 2013].

$$\mathbf{h} = f_{\text{word}}(w_{i-1}) + f_{\text{word}}(w_{i-2}) + \dots + f_{\text{word}}(w_{i-n+1}) \quad (5.3)$$

2. Recurrent neural network (RNN):

A RNN-based language model feeds the words in the context in series into the network, where the context-vector is computed as the concatenation of the word-vector and another context-vector of the older context. [Mikolov et al., 2011a].

$$\mathbf{h}_{i-1} = \langle f_{\text{context}}(\mathbf{h}_{i-2}), f_{\text{word}}(w_{i-1}) \rangle \quad (5.4)$$

The context-vector is computed recursively, as follows:

$$\mathbf{h}_{i-2} = \langle f_{\text{context}}(\mathbf{h}_{i-3}), f_{\text{word}}(w_{i-2}) \rangle$$

$$\mathbf{h}_{i-3} = \langle f_{\text{context}}(\mathbf{h}_{i-4}), f_{\text{word}}(w_{i-3}) \rangle$$

...

and so on, where $f_{\text{context}}(\cdot)$ is a non-linear mapping function.

The RNN-based language model has been improved by incorporating a gating unit in the network so that distant words can be more effectively managed by the model, e.g. the LSTM model [Sundermeyer et al., 2012].

In general, the neural network language models have been empirically shown to outperform the count-based models in terms of perplexities and WER in speech recognition tasks. Also, as the RNN-based models utilize more complex non-linear function to compute the context-vector, they have been shown to outperform the FNN-based models [Mikolov, 2013].

5.2 Neural Network based TDTO Model

In this section, the neural network implementation of the TDTO model is described. The intention is to take advantage of the superior smoothing property of the neural networks

to obtain more accurate probabilities from the TDTO model. A feedforward neural network (FNN) is used for the implementation, where the TD and TO events in the context are encoded to be fed to the FNN to output the probability distribution of words. The neural network based TDTO model, referred to as the NN-TDTO model, is evaluated in terms of perplexity and compared to the discrete TDTO model implementation discussed in Chapter 3 and 4. Also, the comparison will be extended to an automatic speech recognition (ASR) task to evaluate the NN-TDTO model performance.

5.2.1 Motivations

In general, the advantages of using neural networks to implement the TDTO model are three-fold:

1. **Tackling the data scarcity problem:** As inspired by the FNN-based language model, “similar” instances of TD and TO are expected to be projected close to each other in the continuous space. For example, the TDs that correspond to word-pair “*collapse-market*” of different distances shall reside in a neighbor subspace as content words are generally less dependent on their locations in a context. Hence, a word-pair “*collapse-market*” with an unseen distance will be assigned a probability “close” to the same word-pair but of other distances.
2. **Recovering inter-dependencies between words:** As discussed in Section 3.2.2, words in the history are commonly assumed conditional independent to reduce the model parameters. For neural network implementation, as the parameters can be learned in a simpler manner, i.e. back-propagation, such relaxation is not required. The instances of TD and TO in a given context can be fed instantly to a neural network as to retain the inter-dependencies between the words.
3. **Avoiding hyper-parameter tuning in the TDTO model:** Applying the TDTO model requires tuning the interpolation weights and the parameters in the

moving average filter and back-off mechanism. For neural network implementation, such parameter tuning is not needed.

5.2.2 Architecture of the NN-TDTO model

A feedforward neural network (FNN) is used for the implementation of the TDTO model, where the TD and TO in the history of a given target-word are encoded into the input of the FNN to output the probability distribution of words. This model is referred to as the neural network based TDTO model (NN-TDTO) and the network architecture is shown in Figure 5.1.

Figure 5.1: Neural network architecture for the implementation of the NN-TDTO language model.

Each layer in the FNN is described as follows.

Input layer

At the input layer, the TD and TO events in the context are encoded into two vectors to give descriptions about the distances and occurrences of words, respectively. Each

coding vector has a dimension of N , i.e. vocabulary size, where the j -th element in the vector corresponds to the j -th word in the vocabulary. Each coding vector is described as follows:

1. **The TD coding vector:** It is used to describe the positions of words in the context. It is assumed that the correlation decays with respect to the distance of words, the TD encoding scheme is implemented as a reciprocal function:

$$z_j^D(w_{i-k}) = \begin{cases} \frac{1}{k^a} & \text{if } w_{j-k} = v_j \\ 0 & \text{otherwise} \end{cases} \quad (5.5)$$

where $\mathbf{z}_j^D$ 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, it's assumed $k = \infty$ thus $z_j^D = 0$.

For example, given the word *collapse* is located in the 6-th position in the context, the corresponding element in the TD coding vector will be assigned $\frac{1}{6^a}$.

2. **The TO encoding vector:** It is used to describe the occurrences of words in the context according to the following encoding scheme:

$$z_j^O(w_{i-k}) = \begin{cases} \#(O(w_{i-k})) & \text{if } w_{j-k} = v_j \\ 0 & \text{otherwise} \end{cases} \quad (5.6)$$

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

For example, given the word *collapse* occurs only once in the context, the corresponding element in the TO coding vector will be assigned 1.

Essentially, the TD and TO coding vectors can be interpreted as the traditional BOW vector, scaled as according to the position and counts information in the context.

First hidden layer

At the first hidden layer, the TD and TO coding vectors, $\mathbf{z}^D$ and $\mathbf{z}^O$, are projected into their respective continuous space, as follows:

$$\mathbf{c}^D = \mathbf{Q}^D \mathbf{z}^D \quad (5.7)$$

$$\mathbf{c}^O = \mathbf{Q}^O \mathbf{z}^O \quad (5.8)$$

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 $P \times N$, where P denotes the dimension of the representation vectors $\mathbf{c}^D$ and $\mathbf{c}^O$, i.e. $P = |\mathbf{c}^D| = |\mathbf{c}^O|$.

Note that the row vectors of $\mathbf{Q}^D$ and $\mathbf{Q}^O$ are essentially the embedding vectors for the words in the vocabulary, derived from the TD and TO information, respectively. The context vectors are obtained by accumulating, i.e. adding, the respective TD and TO embedding vectors.

Second hidden layer

The representation vectors are then concatenated to be fed to the second hidden layer. The role of the second hidden layer is to introduce non-linearity to the model, typically, hyperbolic tangent activation functions are used in language modeling [Schwenk, 2007, Mikolov et al., 2010, Sundermeyer et al., 2012]:

$$\mathbf{d} = \tanh(\mathbf{M}\mathbf{c} + \mathbf{b}) \quad (5.9)$$

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

$$\mathbf{c} = [\mathbf{c}^{D'} \mathbf{c}^{O'}]' \quad (5.10)$$

$\mathbf{M}$ is the weight matrix and $\mathbf{b}$ is the bias term vector of the second hidden layers. The dimensions of the weight matrix and the bias vector are $H \times P$ and $H \times 1$, where H denotes the dimension in the hidden layer, i.e. $H = |\mathbf{d}|$.

Output layer

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

$$\mathbf{o} = \mathbf{V}\mathbf{d} + \mathbf{r} \quad (5.11)$$

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

$$\mathbf{p} = \frac{\exp(\mathbf{o})}{\sum_{l=1}^N \exp(o_l)} \quad (5.12)$$

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

5.2.3 Complexity of the NN-TDTO model

In the NN-TDTO model, the computational complexity associated to computing the probability of a word is determined by the size of the hidden layers and the output layer, shown as follows:

$$2 \times P \times H + H + H \times N + N \quad (5.13)$$

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 original FNN-based model [Schwenk, 2007], which grows with regard to the history length, the NN-TDTo model complexity is constant regardless the size of the history length.

For training, the back-propagation algorithm is used to learn the neural network parameters. During 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 NN-TDTo model has several practical advantages, as compared to the TDTo model. First, probability estimates can be computed faster by feeding the inputs across the network, in contrast to the TDTo model which involves the calculation of the normalization term $Z(h)$, 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 NN-TDTo model does not need further fine tuning once it is trained, as opposed to the TDTo model which requires the component weights to be further optimized.

5.3 Experiments

In this section, the evaluation of the NN-TDTo model is discussed. Particularly, the NN-TDTo model is compared with the baseline TDTo model to assess if the neural network implementation provides better smoothing to the TDTo modeling. Both models are compared in terms of perplexity and the word error rate (WER) on a speech recognition system.

The perplexities of the models were evaluated by using the BLLIP Wall Street Journal (WSJ) corpus. The training set consists of 1.7M words, which was accounted for about one tenth of the 87's subset. Rather than using the entire 87's subset, as reported in the previous chapter, a smaller subset 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 5k words, which are adopted from the Aurora-4 task.

The baseline TDTO model was built following the details in Section 3.2, which can be depicted by the following formula.

$$P_{\text{TDTO}}(t = w_i | h_{\text{TDTO}} = w_1^{i-1}) = \frac{\left(\begin{array}{c} P_{3G}(t = w_i | h_{3G} = w_{i-2}^{i-1})^{\beta_{3G}} \\ \prod_{k=1}^{n_{\text{TD}}-1} P_{\text{TD}-k}(h_{\text{TD}-k} = \Delta(w_{i-k}) | h_{\text{TO}-k} = O(w_{i-k}), t = w_i)^{\beta_{\text{TD}}} \\ \prod_{k=1}^{n_{\text{TO}}-1} P_{\text{TO}-k}(h_{\text{TO}-k} = O(w_{i-k}) | t = w_i)^{\beta_{\text{TO}}} \end{array} \right)}{Z(h_{3G}, h_{\text{TD}-1}, h_{\text{TD}-2}, \dots, h_{\text{TO}-1}, h_{\text{TO}-2}, \dots, \beta_{3G}, \beta_{\text{TD}}, \beta_{\text{TO}})}$$

For the NN-TDTO model, the FNN 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 dimensions of the input and output layers follow the size of the vocabulary, i.e. 5k. The training set was used for conducting the training while the development set was used for validation during the training. The FNN implementation used in the NN-TDTO model was modified from the CSLM toolkit [Schwenk, 2013].

5.3.1 Perplexity Evaluation

The perplexities of the NN-TDTO and TDTO models for history lengths from 1 to 10 were compared. The results are shown in Figure 5.2.

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, which in this case is a trigram model.

As seen in Figure 5.2, the NN-TDTO 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 NN-TDTO 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,

Figure 5.2: The perplexities of the NN-TDTO and TDTO models. As compared to the baseline TDTO model, the NN-TDTO model consistently showed lower perplexities for all considered history lengths. Note that for the context length of one, the comparatively low perplexity of the TDTO model was contributed by the trigram component that considers a history length of 2. In this case, it is not a fair comparison. The relative reduction (Rel. Red.) percentage is depicted in the right vertical axis.

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 NN-TDTO model. The relative reduction depicted in the right vertical axis in Figure 5.2. Overall, the neural network based implementation has shown to provide a better generalization capability with respect to the original TDTO model implementation.

5.3.2 Speech Recognition

The performance of the NN-TDTO model was also compared with the TDTO model in a speech recognition N -best re-ranking task. The intention is to examine if the improved

smoothing property in the TDTO model benefits the speech recognition systems. The experiment was conducted on the clean subset of the Aurora-4 corpus [Parihar et al., 2015]. The speech recognition system, including the acoustic model and the language model, was configured in the same manner as in the experiment depicted in Section 4.2.1. To be more specific, for each utterance, a list of 200 hypotheses was generated by using the HTK’s HVite decoder and then the language model scores were recomputed by using the TDTO and the NN-TDTO 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 NN-TDTO model result. The NN-TDTO 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 5.3.

Figure 5.3: WER results of the NN-TDTO model as compared to the TDTO model. The NN-TDTO model showed lower WER in all considered history lengths, from two to ten. The relative reduction (Rel. Red.) is depicted in the right vertical axis.

The WER results are consistent with the perplexity evaluation results shown in Figure 5.2. For all the considered history lengths, except for history length of one, the NN-

TDTO 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 WER is depicted in the right vertical axis. The results confirm the effectiveness of the continuous space implementation in the NN-TDTO model.

5.3.3 NN-TDTO vs. NN-NG

Also, the NN-TDTO model was compared with the original FNN-based language model [Bengio et al., 2003, Schwenk, 2007]. Typically, FNNs are used to model the n -grams, where $(n - 1)$ words in the history-context are encoded and fed to the FNN to output a distribution of probabilities (see Section 2.2.1). The NN-TDTO model, on the other hand, encodes the history-context as the TD and TO vectors (see Section 5.2.2). Therefore, the main difference of the NN-TDTO model from the original implementation, referred to as the NN-NG model, lies in the representation of the history-context in the input layer.

The models were compared in terms of perplexities, for history-context of length 1 to 10. The results are shown in Figure 5.4.

As shown in Figure 5.4, the NN-TDTO model did not perform significantly different from the NN-NG model. This observation shows that the TD and TO can be alternatively used to represent the word sequence in the context. In fact, as the dimension of the projection layer was increased, the NN-TDTO model consistently showed lower perplexities than the NN-NG model, particularly for longer context. Figure 5.5 shows the perplexities of the models of projection dimensions 100 and 150. Furthermore, the NN-TDTO model was shown to be more resistant to the noise incurred by the far context, i.e. length >7 , as compared to the NN-NG model which showed increasing perplexity for longer context, i.e. length >10 . Such advantage may be contributed by the TD encoding scheme, see Equation (5.7), that tends to omit the positions of words in the far history-context, which are generally noisy.

The average per-epoch training time of both models was also examined. In Section 5.2.2, the time complexity of the NN-TDTO model has been shown to be constant with

Figure 5.4: The perplexities of the NN-TDTO and NN-NG models for history lengths from one to ten. The performances of both models are not significantly different.

respect to the length of the considered history, as opposed to the NN-NG model that increases linearly, see Equation (5.13). This advantage is further verified empirically. The results are shown in Figure 5.6.

As seen in the figure, the per-epoch training time of the NN-TDTO model was generally unchanged with respect to the length of the history-context, as compared to the case of the NN-NG model which increased almost linearly. In general, for n -grams orders greater than three, the NN-TDTO has been shown to be a more efficient approach. The slight increase of the NN-TDTO training time in higher orders was likely due to the linear combination operation which involved more TD and TO coding-vector, see Equation (5.5) and (5.6). The results shown in Figure 5.6 directly indicate that the NN-TDTO model possesses a faster evaluation time, which is a critical aspect to many real-time applications.

Although the NN-TDTO model gains a more efficient computation time, it requires

Figure 5.5: The perplexities of the NN-TDTO and NN-NG models with the projection layers’s dimensions increased to 100 (top) and 150 (bottom). Although both models did not lower perplexities for history length exceeds seven, the NN-TDTO model is more resistant to the noise incurred by the long context.

comparatively higher memory consumption in lower n -gram orders. For the NN-NG model, all history-words share one common projection matrix; while the NN-TDTO requires two projection matrices for the TD and the TO, which demands an additional memory of size $P \times N$. However, the architecture of the NN-NG model becomes more complex as n grows larger. In this experiment, for example, the NN-NG model consumed more memory than the NN-TDTO for n -gram orders greater than eleven.

5.4 Summary

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

Figure 5.6: the average per-epoch training time results of the NN-TDTO and NN-NG models. The training time of the NN-TDTO model did not change significantly with respect to the history length, as opposed to the NN-NG model’s training time that increased linearly.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis has presented a novel approach to exploiting the long context for language modeling. The proposed model, coined as the term-distance term-occurrence (TDTO) model, estimates probabilities based on the events of term-distance (TD) and term-occurrence (TO), which correspond to the positions and occurrences of words in the history-context. This thesis addresses the weakness in the conventional approaches to combining language models, where the inter-dependencies among the events exploited by the component models are neglected.

In Chapter 3, the formulation of the TDTO model has been presented: the joint event of word sequence and bag-of-words (BOW) is decoupled into the TD and TO events, each of the events is then modeled separately as a different component model. The advantage of the decoupling operation is two-fold:

1. The inter-dependencies between the TD and the TO events are retained after decoupling and can be captured by the language model. Traditional techniques for combining language models such as the linear interpolation (LI) and the log-linear interpolation (LLI) usually neglect the inter-dependencies between events of different types from a given context.

2. The data scarcity problem is reduced by exploiting contexts with two different component models, i.e. the TD and the TO components, where the TO component is used to exploit coarser event, i.e. the binary occurrences of words, which is less susceptible to the data scarcity problem. The contribution of these two components towards the final prediction can be tuned to adapt to different training conditions, e.g. different amount of training data, different domains, etc.

In Chapter 4, the performance on the TDTO model has been evaluated. For model perplexity, as evaluated on the BLLIP Wall Street Journal (WSJ) and the Switchboard (SWB) corpora, the TDTO model showed being capable to exploit the distant context, i.e. up to history length of seven words, which is beyond the reach of a state-of-the-art n -gram model. The TDTO model has shown to reduce trigram perplexities up to 11.2% and 6.5% for the cases of the WSJ and the SWB corpora, respectively. For higher orders of n -gram models, the TDTO model consistently reduces model perplexities. Also, the TDTO model has been shown to outperform other traditional approaches to exploiting the distant context, such as distant-bigram, trigger, and BOW models. Such findings have been shared with the linguistic and the speech processing communities in the following three publications:

Chong, T.Y., Banchs, R.E., Chng, E.S. & Li, H., 2013, Modeling of term-distance and term-occurrence information for improving n-gram language model performance, in Proc. ACL, pp. 233–237.

Chong, T.Y., Banchs, R.E., Chng, E.S. & Li, H., 2014, Improving language modeling by using distance and co-occurrence information of word-pairs and its application to LVCSR, in Proc. ICASSP, pp. 4883–4887.

Chong, T.Y., Banchs, R.E., Chng, E.S. & Li, H., 2015, Decoupling word-pair distance and co-occurrence information for effective long history context language modeling, Trans. TASLP, 23(7): 1221-1232.

In Chapter 5, the problem of smoothing the TDTO model has been addressed by means of a neural network implementation of the TDTO model. The presented implementation, which is referred to as NN-TDTO, is based on a feedforward neural network architecture. Due to the superior smoothing capabilities offered by neural networks, the NN-TDTO model outperformed the TDTO baseline model in both perplexity evaluation and speech recognition re-ranking task. Evaluated on the WSJ corpus, the NN-TDTO model yielded up to 9.2% lower perplexity as compared to the TDTO model. In the Aurora-4 corpus, the NN-TDTO model obtained up to 12.9% relatively lower WER. The finding has been published in the following paper:

Chong, T.Y., Banchs, R.E., Chng, E.S. & Li, H., 2015, TDTO language modeling with feedforward neural networks, in Proc. INTERSPEECH, pp. 233–237.

6.2 Future Work

The key benefit of the TDTO model lies on the decoupling operation, which decomposes the word sequence in a given context into components referred to as Term Distance (TD) and Term Occurrence (TO). While the TO component pays attention to the occurrences of words in the context, which has been addressed by many other approaches including LSA, pLSA, and LDA, the TD component is a novel feature introduced by the TDTO modeling. The TD component pays attention to the specific word positions given the occurrences of the respective words in the context. As future work, the modeling of the TD component is a key direction to further research in the TDTO model framework formulation.

6.2.1 Class-based TD

The TD component intends to model information about the specific location of words in the context, which mainly determines information of syntactic nature in the context. Currently, a moving average filter is used as smoothing mechanism for the TD component

in the TDTO model formulation, and a feedforward neural network (FNN) is applied in the case of the NN-TDTO model formulation. Although significant improvement has been achieved with by the NN-TDTO model, the approach still lacks of linguistic background, i.e. it has the deficiency of not exploiting possible syntactic constraints among words in natural language. For example, some selected word-pairs, due to their POSs, may exhibit more consistent distances than others.

For future work, the use of POS information into the modeling and the smoothing of the TD component should be explored. For example, the distance between two function words, such as *the-of* should be more consistent than the distance between two content words, such as *collapse-stocks*, or mixture of function and content words, such as *collapse-the*. As such, the weight of the running average filter has to be a function that depends on the POSs of the considered words.

6.2.2 Convolutional Neural Network based TDTO Model

The convolutional neural network (CNN) [Dauphin et al., 2016b, Pham et al., 2016b] has been recently applied to language modeling and has been shown to outperform both the feedforward neural network (FNN) and the recurrent neural network (RNN) language models. A CNN-based language model includes additional convolutional layers preceding the conventional FNN. The main function of these additional layer is to construct new features from the input space that allow for more accurate predictions.

As for the TDTO model, the idea of extracting new categories of features motivates exploring a new implementation by using the CNN framework.

Bibliography

- [Antonie and Zaiane, 2002] Antonie, M.-L. and Zaiane, O. (2002). Text document categorization by term association. *Icdm*, pages 19–26.
- [Bacchiani et al., 2006] Bacchiani, M., Riley, M., Roark, B., and Sproat, R. (2006). MAP adaptation of stochastic grammars. *Computer Speech and Language*, 20(1):41–68.
- [Bahl et al., 1983] Bahl, L., Jelinek, F., and Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions On Pattern Analysis and Machine Intelligence*.
- [Bahl et al., 1977] Bahl, L. R., Baker, J. K., Jelinek, F., and Mercer, R. L. (1977). Perplexity—a measure of the difficulty of speech recognition tasks. *Journal of The Acoustical Society of America*, 62(S1):63.
- [Bahl et al., 1989] Bahl, L. R., Brown, P. F., Souza, P. V., and Mercer, R. L. (1989). A Tree-Based Statistical Language Model for Natural Language Speech Recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(7).
- [Beeferman et al., 1997] Beeferman, D., Berger, A., and Lafferty, J. (1997). A model of lexical attraction and repulsion. *Proceedings of the 35th annual meeting on Association for Computational Linguistics -*, pages 373–380.
- [Bellegarda, 1998] Bellegarda, J. R. (1998). A Multispan language modeling framework for large vocabulary speech recognition. *IEEE Transactions on Speech and Audio Processing*, VOL. 6, NO. 5, 6(5):456–467.

- [Bengio et al., 2003] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155.
- [Bermingham and Smeaton, 2010] Bermingham, A. and Smeaton, A. F. (2010). Classifying Sentiment in Microblogs: Is Brevity an Advantage? *Proceedings of the 19th ACM international conference on Information and knowledge management - CIKM '10*, page 1833.
- [Blei et al., 2002] Blei, D., Ng, A., and Jordan, M. (2002). Latent Dirichlet allocation.
- [Bo-June Hsu, 2007] Bo-June Hsu (2007). Generalized linear interpolation of language models. *2007 IEEE Workshop on Automatic Speech Recognition & Understanding (ASRU)*, pages 136–140.
- [Brown et al., 1990] Brown, P., Cocke, J., Pietra, V. D., Pietra, S. D., Jelinek, J., Lafferty, J., Mercer, R., and Roossina, P. (1990). A statistical approach to Machine Translation. *Computational Linguistics*, 16(2):79–85.
- [Brown et al., 1992] Brown, P. E., Pietra, V. J. D., Mercer, R. L., Pietra, S. a. D., and Lai, J. C. (1992). An estimate of an upper bound for the entropy of English. *Computational Linguistics*, 10598(1):31–40.
- [Cao et al., 2005] Cao, G., Nie, J. Y., and Bai, J. (2005). Integrating word relationships into language models. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 298–305.
- [Charniak, 2001] Charniak, E. (2001). Immediate-Head Parsing for Language Models. *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, (3):124–131.

- [Charniak et al., 2000] Charniak, E., Blaheta, D., Ge, N., Hall, K., Hale, J., and Johnson, M. (2000). Bllip 1987-89 wsj corpus release 1. In *Linguistic Data Consortium, Philadelphia*.
- [Chelba and Jelinek, 2000] Chelba, C. and Jelinek, F. (2000). Structured language modeling. *Computer Speech & Language*, 14(4):283–332.
- [Chen and Goodman, 1996] Chen, S. F. and Goodman, J. (1996). An empirical study of smoothing techniques for language modeling. In *ACL*, volume 13, pages 310–318.
- [Church and Hanks, 1989] Church, K. W. and Hanks, P. (1989). Word association norms, mutual information, and lexicography. *Proceedings of the 27th annual meeting on Association for Computational Linguistics -*, pages 76–83.
- [Coccaro, 2005] Coccaro, N. B. (2005). *Latent Semantic Analysis as a Tool to Improve Automatic Speech Recognition Performance*. PhD thesis.
- [Collins, 2003] Collins, M. (2003). {Head-Driven} Statistical Models for {N}atural {L}anguage Parsing. *Cl*, 29(4).
- [Dauphin et al., 2016a] Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. (2016a). Language Modeling with Gated Convolutional Networks. *arX*, page 1612.08083v1 [cs.CL].
- [Dauphin et al., 2016b] Dauphin, Y. N., Fan, A., Auli, M., and Grangier, D. (2016b). Language modeling with gated convolutional networks.
- [Duan and Hsu, 2011] Duan, H. and Hsu, B.-J. P. (2011). Online spelling correction for query completion. *Proceedings of the 20th international conference on World wide web*, pages 117–126.
- [Dumais et al., 1998] Dumais, S., Platt, J., Heckerman, D., and Sahami, M. (1998). Inductive Learning Algorithms and Representations for Text Categorization. *Proceeding*

- of the 1998 ACM 7th International Conference on Information and Knowledge Management, pages 148–155.
- [Fürnkranz, 1998] Fürnkranz, J. (1998). A Study Using n -gram Features for Text Categorization. *Austrian Research Institute for Artificial Intelligence*, (1998):1–10.
- [Gao et al., 2001] Gao, J., Goodman, J. T., and Miao, J. (2001). The Use of Clustering Techniques for Language Modeling – Application to Asian Languages. *Computational Linguistics and Chinese Language Processing*, 6(1):27—60.
- [Godfrey and Holliman, 1997] Godfrey, J. J. and Holliman, E. (1997). Switchboard-1 Release 2. In *Linguistic Data Consortium, Philadelphia*.
- [Good, 1953] Good, I. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3/4):237–264.
- [Guthrie et al., 2006] Guthrie, D., Allison, B., and Liu, W. (2006). A closer look at skip-gram modelling. *L*, pages 1222–1225.
- [Gutkin, 2000] Gutkin, A. (2000). *Log-Linear Interpolation of Language Models*. PhD thesis, University of Cambridge.
- [Heafield et al., 2016] Heafield, K., Geigle, C., Massung, S., and Schwartz, L. (2016). Normalized Log-Linear Interpolation of Backoff Language Models is Efficient. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 876–886.
- [Hofmann, 1999] Hofmann, T. (1999). Probabilistic Latent Semantic Analysis. *UAI '99: Proceedings of Uncertainty in Artificial Intelligence*, pages 289–296.
- [Huang et al., 1993] Huang, X., Allea, F., Hon, H.-W., Hwang, M.-Y., and Rosenfeld, R. (1993). The SPHINX-II speech recognition system: an overview. *Computer Speech and Language*, 7(2):137–148.

- [Jelinek, 1990] Jelinek, F. (1990). Self-organized language modeling for speech recognition. *Readings in Speech Recognition*, pages 450–506.
- [Jelinek and Mercer, 1980] Jelinek, F. and Mercer, R. L. (1980). Interpolated estimation of Markov source parameters from sparse data. *Proceedings of the Workshop on Pattern Recognition in Practice*, pages 381–397.
- [Katz, 1987] Katz, S. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401.
- [Kim, 2014] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification.
- [Klakow, 1998] Klakow, D. (1998). Log-linear interpolation of language models. In *ICSLP*, number January, pages 1–4.
- [Kneser and Ney, 1995] Kneser, R. and Ney, H. (1995). Improved backing-off for M-gram language modeling. *ICASSP*, 1:181–184.
- [Lau et al., 1993a] Lau, R., Rosenfeld, R., and Roukos, S. (1993a). Adaptive language modeling using the maximum entropy principle. In *Human Language Technology*.
- [Lau et al., 1993b] Lau, R., Rosenfeld, R., and Roukos, S. (1993b). Trigger-based language models: a maximum entropy approach. In *ICASSP*, pages 45–48.
- [Liu et al., 2013] Liu, X., Gales, M. J., and Woodland, P. C. (2013). Use of contexts in language model interpolation and adaptation. *Computer Speech and Language*, 27(1):301–321.
- [Manning and Carpenter, 1997] Manning, C. D. and Carpenter, B. (1997). Probabilistic Parsing Using Left Corner Language Models. page 12.
- [Marti and Bunke, 2001] Marti, U.-V. and Bunke, H. (2001). Using a Statistical Language Model To Improve the Performance of an Hmm-Based Cursive Handwriting

- Recognition System. *International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI)*, 15(01):65–90.
- [Matiasek and Baroni, 2003] Matiasek, J. and Baroni, M. (2003). Exploiting Long Distance Collocational Relations in Predictive Typing. *Proceedings of the EACL Workshop on Language Modeling for Text Entry Methods*, pages 1–8.
- [Mikolov, 2013] Mikolov, T. (2013). Efficient Estimation of Word Representations in Vector Space.
- [Mikolov et al., 2011a] Mikolov, T., Deoras, A., Povey, D., Burget, L., and Černocký, J. (2011a). Strategies for training large scale neural network language models. In *ASRU*, pages 196–201.
- [Mikolov et al., 2010] Mikolov, T., Karafiat, M., Burget, L., Honza, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *Interspeech*.
- [Mikolov et al., 2011b] Mikolov, T., Kombrink, S., Burget, L., Černocký, J., and Khudanpur, S. (2011b). Extensions of recurrent neural network language model. In *ICASSP*, pages 5528–5531.
- [Nádas, 1984] Nádas, A. (1984). Estimation of probabilities in the language model of the IBM speech recognition system. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(4):859–861.
- [Ney et al., 1994] Ney, H., Essen, U., and Kneser, R. (1994). On structuring probabilistic dependences in stochastic language modelling. *Computer Speech & Language*, 8(1):1–38.
- [Novak and Mammone, 2001] Novak, M. and Mammone, R. (2001). Use of non-negative matrix factorization for language model adaptation in a lecture transcription task. *ICASSP*, pages 541–544.

- [Pang and Lee, 2004] Pang, B. and Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 271.
- [Pang et al., 2002] Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up? *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - EMNLP '02*, 10:79–86.
- [Parihar et al., 2015] Parihar, N., Picone, J., Pearce, D., and Hirsch, H. G. (2015). Performance analysis of the Aurora large vocabulary baseline system. In *European Signal Processing Conference*, volume 06-10-Sept, pages 553–556.
- [Peng and Schuurmans, 2003] Peng, F. and Schuurmans, D. (2003). Combining Naive Bayes and n-Gram Language Models for Text Classification. In *25th European Conference on Information Retrieval Research (ECIR)*, pages 335–350.
- [Pham et al., 2016a] Pham, N.-q., Kruszewski, G., and Boleda, G. (2016a). Convolutional Neural Network Language Models. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP2016)*, pages 1153–1162.
- [Pham et al., 2016b] Pham, N.-Q., Kruszewski, G., and Boleda, G. (2016b). Convolutional neural network language models. In *EMNLP*, pages 1153–1162.
- [Ponte and Croft, 1998] Ponte, J. M. and Croft, W. B. (1998). A language modeling approach to information retrieval. *Scholarworks.Umass.Edu*.
- [Potamianos and Jelinek, 1998] Potamianos, G. and Jelinek, F. (1998). A study of n-gram and decision tree letter language modeling methods. *Speech Communication*, 24:171–192.
- [Roark et al., 2010] Roark, B., Villiers, J. D., Gibbons, C., and Fried-oken, M. (2010). Scanning methods and language modeling for binary switch typing. *Proceedings of*

- the NAACL HLT 2010 Workshop on Speech and Language Processing for Assistive Technologies*, (June):28–36.
- [Rosa, 2002] Rosa, J. L. G. (2002). Next word prediction in a connectionist distributed representation system. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 3:596–601.
- [Rosenfeld, 1996] Rosenfeld, R. (1996). A maximum entropy approach to adaptive statistical language modelling. *Computer Speech & Language*, 10(3):187–228.
- [Samuelsson and Reichl, 1999] Samuelsson, C. and Reichl, W. (1999). A class-based language model for large-vocabulary speech recognition extracted from part-of-speech statistics. *Icassp*, pages 537–540.
- [Schwenk, 2007] Schwenk, H. (2007). Continuous space language models. *Computer Speech and Language*, 21(3):492–518.
- [Schwenk, 2013] Schwenk, H. (2013). CSLM - A modular open-source continuous space language modeling toolkit. In *Interspeech*, number 6, pages 1198–1202.
- [Simons et al., 1997] Simons, M., Ney, H., and Martin, S. (1997). Distant bigram language modelling using maximum entropy. *ICASSP*, 2:787–790.
- [Siu and Ostendorf, 2000] Siu, M. and Ostendorf, M. (2000). Variable n-gram and extensions for conversational speech language modeling. *IEEE Transactions on Speech and Audio Processing*, 8(1):63–75.
- [Song and Croft, 1999] Song, F. and Croft, W. B. (1999). A general language model for information retrieval. *Eighth International Conference on Information and Knowledge Management CIKM 99*, pages 316–321.
- [Steffen, 2004] Steffen, J. (2004). N-Gram Language Modeling for Robust Multi-Lingual Document Classification. *Proceedings of the 4th International Conference on Language Resources and Evaluation.*, pages 731–734.

- [Sundermeyer et al., 2012] Sundermeyer, M., Schl, R., and Ney, H. (2012). LSTM neural networks for language modeling. In *Interspeech*, pages 194–197.
- [Thiele et al., 2000] Thiele, F., Rueber, B., and Klakow, D. (2000). Long range language models for free spelling recognition. *2000 Ieee International Conference on Acoustics, Speech, and Signal Processing, Proceedings, Vols I-Vi*, pages 1715–1718\3906.
- [Van Uytsel et al., 2001] Van Uytsel, D. H., Van Aelten, F., and Van Compernelle, D. (2001). A structured language model based on context-sensitive probabilistic left-corner parsing. *Second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies 2001 - NAACL '01*, pages 1–8.
- [Vatanen et al., 2010] Vatanen, T., Jaakko, J. V., and Virpioja, S. (2010). Language Identification of Short Text Segments with N-gram Models. *Lrec 2010*, pages 3423–3430.
- [Wandmacher and Antoine, 2008] Wandmacher, T. and Antoine, J.-Y. (2008). Methods to integrate a language model with semantic information for a word prediction component. *Computational Linguistics*, (June):10.
- [Witten and Bell, 1991] Witten, I. H. and Bell, T. C. (1991). The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094.
- [Wu and Wang, 2012] Wu, M. S. and Wang, H. M. (2012). A term association translation model for naive Bayes text classification. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7301 LNAI(PART 1):243–253.
- [Xu and Jelinek, 2004] Xu, P. and Jelinek, F. (2004). Random Forests in Language Modeling. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

- [Young et al., 2006] Young, S., Evermann, G., and Gales, M. (2006). *The HTK book version 3.4*.
- [Zhai and Lafferty, 2004] Zhai, C. and Lafferty, J. (2004). A Study of Smoothing Methods for Language Models Applied to Information Retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214.
- [Zhou et al., 2002] Zhou, Z., Gao, J., and Chang, E. (2002). Improving language modeling by combining heterogeneous corpora. . . . *on Chinese Spoken Language*