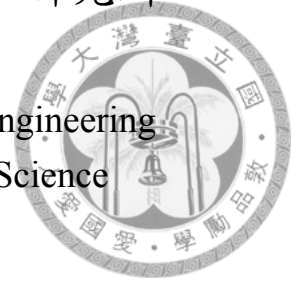


國立臺灣大學電機資訊學院資訊工程學研究所  
博士論文

Department of Computer Science and Information Engineering  
College of Electrical Engineering and Computer Science  
National Taiwan University  
Doctoral Dissertation



語意理解與語句生成之對偶性的有效利用  
Exploiting the Duality between Language Understanding  
and Generation and Beyond

蘇上育  
Shang-Yu Su

指導教授：陳縉儂博士  
Advisor: Yun-Nung Chen, Ph.D.

中華民國 110 年 10 月  
October, 2021





# Acknowledgements

First and foremost, my greatest thanks go to my advisor, and also my friend, Yun-Nung (Vivian) Chen. Before meeting Vivian, I have never thought about pursuing a PhD degree. It was Vivian showing her passion of giving back to her alma mater and home country. More importantly, Vivian is an extremely kind and supportive advisor that I could not have asked for more. I can literally talk with her about everything. Training me to be a brilliant researcher is not her ultimate goal, I have barely received demands for research outputs from her. She did not teach me NLP and how to do research by holding my hands, instead, she gave me full freedom to explore research and life and develop my own philosophy of life. Thank you Vivian, it is really my great honor to be your first PhD student, and it is my great luck to have you on my side during this journey. I appreciate your steady support and trust, thank you for more than I can describe. Pursuing the PhD degree is beautiful serendipity and has already changed my life.

My path to the degree would not have been this smooth without the help from all the people I met during the journey, I am not attempting to name everyone here since I know I would miss some. But I sincerely appreciate friendship, technical discussions from you guys. First, I would like to thank MiuLab members for all the accompanies and time we spent together: Chih-Wen Goo, Guang-He Lee, Ting-Yun Chang, Ta-Chung Chi, Ting-Rui Chiang, Jian-Jia Su, Tze-Teng Wong, Yi-Ting Yeh, Yi-Hsuan Deng, Yu-An Wang, Hao-Tong Ye, Chao-Wei Huang, Shang-Chi Tsai, Kai-Lin Lo, Tzu-Chuan

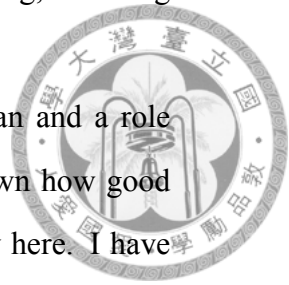
Lin, Hsiao-Hua Cheng, Shan-Wei Lin, Kuang Kao, Yu-Kai Huang, Yen-Ting Lin, Yung-Sung Chuang, Po-Wei Lin, and Ting-Chun Wang.

Special thanks to Prof. Hung-Yi Lee, you are a superman and a role model for all the CS PhD students in Taiwan. You have shown how good a PhD/researcher from NTU could be, giving us hopes to stay here. I have learned so much from you, and I believe there are countless students are just like me. Writing a book chapter with you was a very unique and rewarding experience for me.

During my PhD, I have done wonderful internships at Microsoft, Amazon Alexa, and Google Research. I would like to express my gratitude to all the mentors and colleagues I met during the internships. My experience at Microsoft helped me get started on research and stay on the path, thank you XiuJun Li, Jianfeng Gao, and Jingjing Liu. Thank Amazon Alexa for broadening my research and enabling collaborations with the great researchers, thank you Mihail Eric, Behnam Hedayatnia, Dilek Hakkani-Tür, Han Li, Lei Shu, Sungjin Lee, and Sunghyun Park. Thank Abhinav Rastogi for hosting me at Google Research, working with and learning from you was truly an awesome experience. I also would like to thank Google for providing me with the fellowship.

Thank my friends and classmates from NTU EE for taking care of me. When I was interning in the states, these friends always proactively invited loner me to hang out. Without you guys, my internships would not have been this fun. If I did not start doing research, I would never have a chance to meet awesome people from all over the world: Jiho Park, Jamin Shin, Nayeon Lee, Hwaran Lee, Seonil Son, Ting-Yao Hsu, Chieh-Yang Huang, Wei-Ning Hsu, Po-Sen Huang, Yuqin Guo, Qing Ping, Chien-Wei Lin, Alice Chuang, Wenhui Chen, Yuexin Wu, Seokhwan Kim, and more and more. I look forward to meeting you somewhere in the world in the near future.

Thank my old friends, especially Ada, Lynn, PJ, Chung-Hsuan, for help-



ing me get through frustrations and anxiety and sharing the joyful moments.

I sincerely thank IZ\*ONE for pulling me out from the stressful life, bringing me laughter day by day. Last but not least to my dearest family, Mom, Dad, Aunt Mi, and Mimi, thank all of you for your unconditional love and support.

Without any of you none of my success would be possible.







## 摘要

許多現實世界的人工智慧問題都帶有對偶性質，也就是說，我們可以直接調換一個任務的輸入和目標來形成另外一個任務。機器翻譯是一個很經典的例子，舉例來說，從英文翻譯至中文有一個對偶任務為從中文翻譯至英文。語音辨識和語音合成之間也有結構對偶性。給定一個資訊文本的片段，回答問題和產生問題也是對偶態。最近的研究於有效利用任務之間的對偶性來提升表現也顯現了對偶性的重要性。

自然語言理解和自然語言生成皆為自然語言處理以及對話領域的重要研究主題，自然語言理解的目標是抽取出給定語句的核心語意，而自然語言生成則相反，其目標是為基於給定的語意建構對應的句子。然而，語言理解和語言生成之間的對偶性尚未被探討過。

本篇論文旨在探究自然語言理解和自然語言生成之間的結構對偶性。在本篇論文中，我們展示五篇連續的研究，每一篇聚焦在學習以及資料情境的不同層面。第一，我們有效利用了自然語言理解和自然語言生成之間的對偶性並將其作為正則化項導入學習目標。此外，我們利用專業知識來設計適合的方法來估計資料分布。第二，我們進一步提出了聯合學習框架，提供了使用不只是監督式學習還有非監督式學習演算法的彈性、且使兩個模型之間能夠順暢流通梯度。第三，我們研究如何利用最大化相互資訊來增強聯合學習框架。上述的研究都是在訓練階段有效利用對偶性，因此最後，我們向前邁進一步、在推論階段以及預訓練後的微調階段利用對偶性。每一篇研究都展示了一個用不同方式來有效利用對偶性的新模型或是學習框架，總合起來，這篇論文探索了有效利用自然語言理解和自然語言生成之間的結構對

偶性的一個新研究方向。

關鍵字：對話系統，自然語言理解，自然語言生成，對偶學習，語意理解，語句生成







# Abstract

Many real-world artificial intelligence tasks come with a *dual* form; that is, we could directly swap the input and the target of a task to formulate another task. Machine translation is a classic example, for example, translating from English to Chinese has a dual task of translating from Chinese to English. Automatic speech recognition (ASR) and text-to-speech (TTS) also have structural duality. Given a piece of informative context, question answering and question generation are in dual form. The recent studies magnified the importance of the duality by boosting the performance of both tasks with the exploitation of the duality.

Natural language understanding (NLU) and natural language generation (NLG) are both critical research topics in the NLP and dialogue fields. The goal of natural language understanding is to extract the core semantic meaning from the given utterances, while natural language generation is opposite, of which the goal is to construct corresponding sentences based on the given semantics. However, the dual property between understanding and generation has been rarely explored.

This main goal of this dissertation is to investigate the structural duality between NLU and NLG. In this thesis, we present four consecutive studies, each focuses on different aspects of learning and data settings. First, we exploits the duality between NLU and NLG and introduces it into the learning objective as the regularization term. Moreover, expert knowledge is incorporated to design suitable approaches for estimating data distribution. Second,

we further propose a joint learning framework, which provides flexibility of incorporating not only supervised but also unsupervised learning algorithms and enables the gradient to propagate through two modules seamlessly. Third, we study how to enhance the joint framework by mutual information maximization. Above works exploit the duality in the training stage, hence lastly, we make a step forward to leverage the duality in the inference stage and the finetuning stage after pretraining. Each work presents a new model and learning framework exploiting the duality in different manners. Together, this dissertation explores a new research direction of exploiting the duality between language understanding and generation.

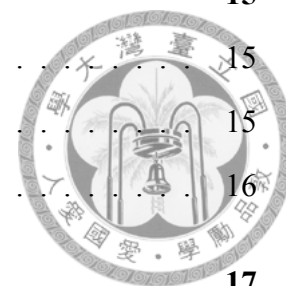
**Keywords:** dialogue systems, natural language understanding, natural language generation, dual learning



# Contents

<b>Acknowledgements</b>	<b>iii</b>
摘要	vii
<b>Abstract</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Contributions . . . . .	2
1.3 Thesis Structure . . . . .	2
<b>2 Background</b>	<b>5</b>
2.1 Recurrent Neural Models . . . . .	5
2.1.1 Recurrent Neural Network (RNN) . . . . .	5
2.1.2 Gated Recurrent Unit (GRU) . . . . .	6
2.2 Transformer . . . . .	6
2.2.1 Multi-Head Attention . . . . .	7
2.2.2 Positional Encoding . . . . .	8
2.3 Dialogue Systems . . . . .	9
2.4 Problem Formulation . . . . .	10
2.5 Datasets . . . . .	11
2.5.1 NLG $\rightarrow$ NLU . . . . .	12
2.5.2 NLG $\rightarrow$ NLU (IOB) . . . . .	12
2.5.3 NLU (IOB) $\rightarrow$ NLG . . . . .	13

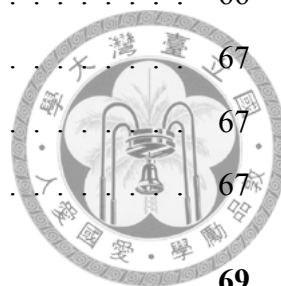
<b>3</b>	<b>Related Work</b>	<b>15</b>
3.1	Natural Language Understanding . . . . .	15
3.2	Natural Language Generation . . . . .	15
3.3	Dual Learning . . . . .	16
<b>4</b>	<b>Dual Supervised Learning</b>	<b>17</b>
4.1	Proposed Framework . . . . .	17
4.1.1	Dual Supervised Learning . . . . .	17
4.1.2	Distribution Estimation as Autoregression . . . . .	19
4.2	Experiments . . . . .	21
4.2.1	Settings . . . . .	22
4.2.2	Model Details . . . . .	22
4.2.3	Results and Analysis . . . . .	23
4.3	Summary . . . . .	24
<b>5</b>	<b>Joint Dual Learning</b>	<b>25</b>
5.1	Proposed Framework . . . . .	26
5.1.1	Joint Dual Learning . . . . .	27
5.1.2	Learning Objective . . . . .	28
5.1.3	Reward Function . . . . .	28
5.1.4	Flexibility of Learning Scheme . . . . .	30
5.2	Experiments . . . . .	32
5.2.1	Model . . . . .	33
5.2.2	Results and Analysis . . . . .	34
5.2.3	Investigation of Hybrid Objectives . . . . .	35
5.2.4	Qualitative Analysis . . . . .	37
5.3	Summary . . . . .	37
<b>6</b>	<b>Dual Mutual Information Maximization</b>	<b>39</b>
6.1	Proposed Method . . . . .	39
6.1.1	Conversations as Transmission of Semantics . . . . .	39



6.1.2	Mutual Information Estimation . . . . .	40
6.1.3	Training . . . . .	42
6.2	Experiments . . . . .	42
6.2.1	Training Details . . . . .	43
6.2.2	Model Structure . . . . .	44
6.2.3	Results and Analysis . . . . .	45
6.3	Summary . . . . .	46
<b>7</b>	<b>Dual Inference</b>	<b>47</b>
7.1	Proposed Framework . . . . .	48
7.1.1	Dual Inference . . . . .	48
7.1.2	Marginal Distribution Estimation . . . . .	49
7.2	Experiments . . . . .	50
7.2.1	Training Details . . . . .	51
7.2.2	Inference Details . . . . .	51
7.2.3	Model Structure . . . . .	52
7.2.4	Results and Analysis . . . . .	52
7.3	Summary . . . . .	55
<b>8</b>	<b>Dual Finetuning</b>	<b>57</b>
8.1	Pretrained Language Models . . . . .	57
8.2	Generative Pretrained Transformer 2 (GPT-2) . . . . .	58
8.3	Objective Design . . . . .	59
8.3.1	De-Training . . . . .	60
8.3.2	Co-Training . . . . .	60
8.3.3	Inference . . . . .	61
8.4	Experiments . . . . .	61
8.5	Summary . . . . .	63
<b>9</b>	<b>Conclusion and Future Work</b>	<b>65</b>
9.1	Challenges . . . . .	65



9.2	Conclusion . . . . .	66
9.3	Future Work . . . . .	67
9.3.1	Advanced Learning Algorithms . . . . .	67
9.3.2	Connection to Pragmatics . . . . .	67
<b>Bibliography</b>		<b>69</b>

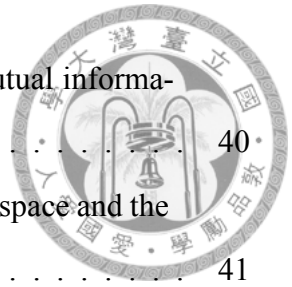




# List of Figures

2.1	The model architecture of Transformer. . . . .	7
2.2	A typical dialogue system pipeline, which is composed of automatic speech recognition (ASR), natural language understanding (NLU), dialogue state tracking (DST), dialogue policy (DP), natural language generation (NLG), and Text-to-Speech (TTS) modules. . . . .	9
2.3	NLU and NLG emerge as a dual form. . . . .	10
2.4	A simple example of using a recurrent unit as NLU model, which we encode an input sentence and use the final hidden state to predict semantic labels. . . . .	11
2.5	A simple example of using a recurrent unit as NLG model, which we feed a semantic vector and the begin-of-sentence (<BOS>) token into the model and predict a text token at each step. . . . .	11
4.1	The illustration of the masked autoencoder for distribution estimation (MADE). . . . .	20
4.2	The NLU model, which we encode an input sentence and use the final hidden state to predict semantic labels. . . . .	23
4.3	The NLG model, which we feed a semantic vector and the begin-of-sentence (<BOS>) token into the model and predict a text token at each step. . . . .	23
5.1	<b>Left:</b> The proposed joint dual learning framework, which comprises <i>Primal Cycle</i> and <i>Dual Cycle</i> . The framework is agnostic to learning objectives and the algorithm is detailed in Algorithm 1. <b>Right:</b> In our experiments, the models for NLG and NLU are a GRU unit accompanied with a fully-connected layer. . . . .	26

5.2	Straight-Through estimator. . . . .	31
6.1	The proposed framework, which we aim to improve the mutual information between the representation of language and semantics. . . . .	40
6.2	The illustration of projections between the natural language space and the semantic space. . . . .	41
6.3	The proposed framework and an analogy of how human converses for one-way transmission of the semantics $x$ . . . . .	41
7.1	The proposed model for estimating the density of a given semantic frame. . . . .	51
8.1	Source: <a href="https://jalammar.github.io/illustrated-gpt2/">https://jalammar.github.io/illustrated-gpt2/</a> . . . . .	58
8.2	The objective design of De-Training, which the model is finetuned for a task at one time. $\langle \text{NLU} \rangle$ , $\langle \text{NLG} \rangle$ , and $\langle \text{EOS} \rangle$ are the special tokens. . . . .	60
8.3	The objective design of Co-Training, which the model is finetuned for both tasks at one time. $\langle \text{NLU} \rangle$ , $\langle \text{NLG} \rangle$ , and $\langle \text{EOS} \rangle$ are the special tokens. . . . .	60
8.4	The illustration of inference of NLU, the input sentence is attached with a $\langle \text{NLU} \rangle$ token to enforce the model to generate semantic tokens according to the given context. . . . .	61
8.5	The comparison of overall performance of finetuning GPT-2 of different sizes, <i>emphde-training</i> is to train a task at one time and <i>co-training</i> is to train both tasks at one time. The results show that Co-Training only works better with smaller models. . . . .	64







# List of Tables

2.1	The statistics of the datasets. . . . .	12
4.1	The NLU performance reported on micro-F1 and the NLG performance reported on BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). . . . .	22
5.1	The NLU performance reported on micro-F and the NLG performance reported on BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). . . . .	33
5.2	An example of the Primal Cycle, where the baseline model is row (a) in Table 5.1. . . . .	35
5.3	An example of the Dual Cycle, where the baseline model is row (a) in Table 5.1. . . . .	35
6.1	For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). All reported numbers are averaged over three different runs. . . . .	44
7.1	The model performance for the ATIS dataset. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). All reported numbers are averaged over three different runs. . . . .	53

7.2	The model performance for the SNIPS dataset. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). All reported numbers are averaged over three different runs. . . . .	53
7.3	The model performance for the E2E NLG dataset. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). All reported numbers are averaged over three different runs. . . . .	54
8.1	The backbones of dual finetuning (Row (d) and (h)) are GPT-2 (Large) and de-training. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). . . . .	62
8.2	The comparison of overall performance of finetuning GPT-2 of different sizes. The default training scheme is de-training, so the rows like Row (a) is the performance of de-training while Row (b) is the performance of co-training. . . . .	63



# Chapter 1

## Introduction

### 1.1 Motivation

Many real-world artificial intelligence tasks come with a *dual* form; that is, we could directly swap the input and the target of a task to formulate another task. Machine translation is a classic example, for example, translating from English to Chinese has a dual task of translating from Chinese to English. Automatic speech recognition (ASR) and text-to-speech (TTS) also have structural duality. Given a piece of informative context, question answering and question generation are in dual form. The recent studies magnified the importance of the duality by boosting the performance of both tasks with the exploitation of the duality.

Natural language understanding (NLU) and natural language generation (NLG) are both critical research topics in the NLP and dialogue fields. The goal of natural language understanding is to extract the core semantic meaning from the given utterances, while natural language generation is opposite, of which the goal is to construct corresponding sentences based on the given semantics. However, the dual property between understanding and generation has been rarely explored.

This main goal of this dissertation is to investigate the structural duality between NLU and NLG. In this thesis, we present five consecutive studies, each focuses on different aspects of learning and data settings. First, we exploits the duality between NLU and NLG and introduces it into the learning objective as the regularization term. Moreover,

expert knowledge is incorporated to design suitable approaches for estimating data distribution. Second, we further propose a joint learning framework, which provides flexibility of incorporating not only supervised but also unsupervised learning algorithms and enables the gradient to propagate through two modules seamlessly. Third, we study how to enhance the joint framework by mutual information maximization. Fourth, since above works exploit the duality in the training stage, hence we make a step forward to leverage the duality in the inference stage. Lastly, we finetune the pretrained language models on the two dual tasks and achieve the goal of solving two dual tasks in a single model. Each work presents a new model and learning framework exploiting the duality in different manners. Together, this dissertation explores a new research direction of exploiting the duality between language understanding and generation.

## 1.2 Contributions

- We were among the first to investigate the duality between natural language understanding and generation.
- We present five consecutive studies, each focuses on different aspects of learning and data settings, including supervised learning with auxiliary objectives, semi-supervised learning, inference, and finetuning.
- Each work presents a new model and learning framework exploiting the duality in different manners, where we focus on both robustness and scalability.

## 1.3 Thesis Structure

The thesis is organized as below.

- Chapter 2 - **Background**

This chapter introduce background knowledge utilized in the proposed methods.

- Chapter 3 - **Related Works**

This chapter reviews related works.

- Chapter 4 - **Dual Supervised Learning**

In this chapter, we first focus on supervised learning and exploit the duality between NLU and NLG and introduces it into the learning objective as the regularization term. This chapter is based on our work [1].



- Chapter 5 - **Joint Dual Learning**

In this chapter, we focus on semi-supervised learning and further propose a joint learning framework, which provides flexibility of incorporating not only supervised but also unsupervised learning algorithms and enables the gradient to propagate through two modules seamlessly. This chapter is based on our work [2].

- Chapter 6 - **Dual Mutual Information Maximization**

In this chapter, we study how to enhance the joint framework by mutual information maximization. We propose to improve the mutual information between language and semantics in training.

- Chapter 7 - **Dual Inference**

Above works exploit the duality in the training stage, hence in this chapter, we make a step forward to leverage the duality in the inference stage. This chapter is based on our work [3].

- Chapter 8 - **Dual Finetuning**

In this chapter, we study how to finetune the pretrained language models on the two dual tasks and achieve the goal of solving two dual tasks in a single model.

- Chapter 9 - **Conclusion and Future Work**

In this chapter, we conclude this thesis and discuss future work and challenges in this research direction.





# Chapter 2

## Background

In this chapter, we first introduce some background knowledge required for this thesis, then describe the problem formulation, and finally detail the data we conducted in the experiments.

### 2.1 Recurrent Neural Models

In this section, we will introduce the standard recurrent neural network (RNN) and Gated Recurrent Unit (GRU) used in the baseline models.

#### 2.1.1 Recurrent Neural Network (RNN)

RNN [4] is designed to capture information of time dimension of a sequential observations  $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T)$ . The network will generate a sequence of hidden representations  $(\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_T)$ , where  $\mathbf{h}_t$  encodes observations  $(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t)$ . The hidden representation is generated recursively like equation 2.1:

$$\mathbf{h}_t = \sigma(\mathbf{W}_h \mathbf{x}_t + \mathbf{U}_h \mathbf{h}_{t-1} + \mathbf{b}_h) \quad (2.1)$$

$\sigma$  is a non-linear activation function applied element-wise (e.g.,  $\tanh$ ) and  $\mathbf{W}_h$ ,  $\mathbf{U}_h$  and  $\mathbf{b}_h$  are the weight and bias to be learned. At each timestep, the hidden state  $\mathbf{h}_t$  is used to predict a target output  $\mathbf{o}_t$  through a linear projection. For example, if we want to predict

the next word in a sentence, the target output  $\mathbf{o}_t$  will be the probabilities of each word in the vocabulary. Formally,

$$\mathbf{o}_t = \text{softmax}(\mathbf{W}_y \mathbf{h}_t + \mathbf{b}_y) \quad (2.2)$$

where  $\mathbf{W}_y$  and  $\mathbf{b}_y$  are parameters to be learned.



### 2.1.2 Gated Recurrent Unit (GRU)

As the observations become longer and longer, the RNN model described in Sec. 2.1.1 will encounter the gradient vanish problem. The problem is that: network uses back propagation to compute gradients. After multiplying numbers smaller than one several times, the “front” time step may receive very small gradient which makes it untrainable. GRU cell [5] is proposed to solve this problem. The internal structure of a GRU cell is:

$$\mathbf{r}_t = \sigma(\mathbf{W}_r \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t]), \quad (2.3)$$

$$\mathbf{z}_t = \sigma(\mathbf{W}_z \cdot [\mathbf{h}_{t-1}, \mathbf{x}_t]), \quad (2.4)$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_{\tilde{h}} \cdot [\mathbf{r}_t * \mathbf{h}_{t-1}, \mathbf{x}_t]), \quad (2.5)$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) * \mathbf{h}_{t-1} + \mathbf{z}_t * \tilde{\mathbf{h}}_t. \quad (2.6)$$

where  $\mathbf{W}_r$  is the parameter for reset gate, and  $\mathbf{W}_z$  is the parameter for update gate. The new hidden state  $\mathbf{h}_t$  is updated by the linear interpolation of original hidden state and transformed hidden state  $\tilde{\mathbf{h}}_t$  weighted by update gate.

## 2.2 Transformer

RNN-based models typically require to generate a sequence of hidden states  $h_t$  as a function of the previous hidden state  $h_{t-1}$  and the input for position  $t$ . This inherently sequential nature precludes parallelization within training examples, which becomes critical at longer sequence lengths. CNN-based models [6], on the other hand, cause difficulties to learn dependencies between distant positions. Recently Transformer [7] is proposed,



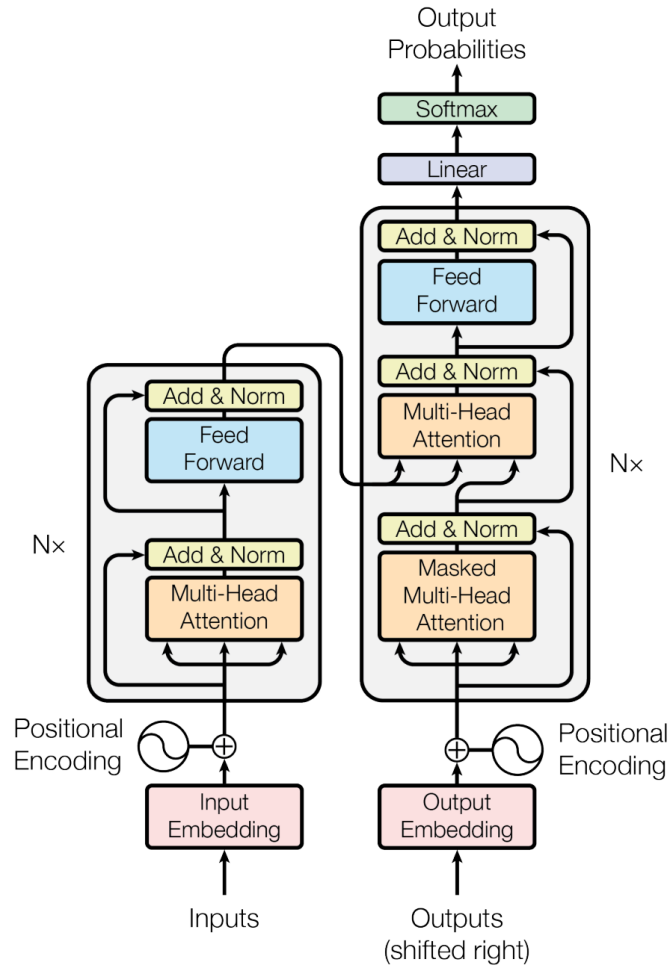


Figure 2.1: The model architecture of Transformer.

which based solely on parallelizable attention mechanisms, dispensing with recurrence and convolutions entirely. The model architecture is illustrated in Figure 2.1.

### 2.2.1 Multi-Head Attention

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

In Transformer, Vaswani et al. [7] proposed multi-head attention, which linearly project the queries, keys and values  $h$  times with different, learned linear projections to  $d_k$ ,  $d_k$  and  $d_v$  dimensions, respectively. On each of these projected versions of queries, keys and val-

ues, the attention function is then performed in parallel, yielding  $d_v$ -dimensional output values. These are concatenated and once again projected, resulting in the final outputs of dimension  $d_{model}$ .

Formally,

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W^O \quad (2.7)$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$

Where the projections corresponding to query, key, value, and the final output are parameter matrices  $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$ ,  $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$ ,  $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$  and  $W^O \in \mathbb{R}^{hd_v \times d_{model}}$ , respectively.

## 2.2.2 Positional Encoding

Since Transformer contains no recurrence and no convolution, Positional Encoding is proposed for the model to make use of the order of the sequence. Instead of using learned embedding, sine and cosine functions are used for positional encoding:

$$PE_{(pos, 2i)} = \sin(pos/10000^{2i/d_{model}}) \quad (2.8)$$

$$PE_{(pos, 2i+1)} = \cos(pos/10000^{2i/d_{model}}), \quad (2.9)$$

where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponds to a sinusoid. The authors hypothesized it would allow the model to easily learn to attend by relative positions, since for any fixed offset  $k$ ,  $PE_{pos+k}$  can be represented as a linear function of  $PE_{pos}$ . The positional encodings have the same dimension  $d_{model}$  as the input embeddings, and are added to the input embeddings at the bottoms of the encoder and decoder stacks to provide the ordering information of each word in the sequence.

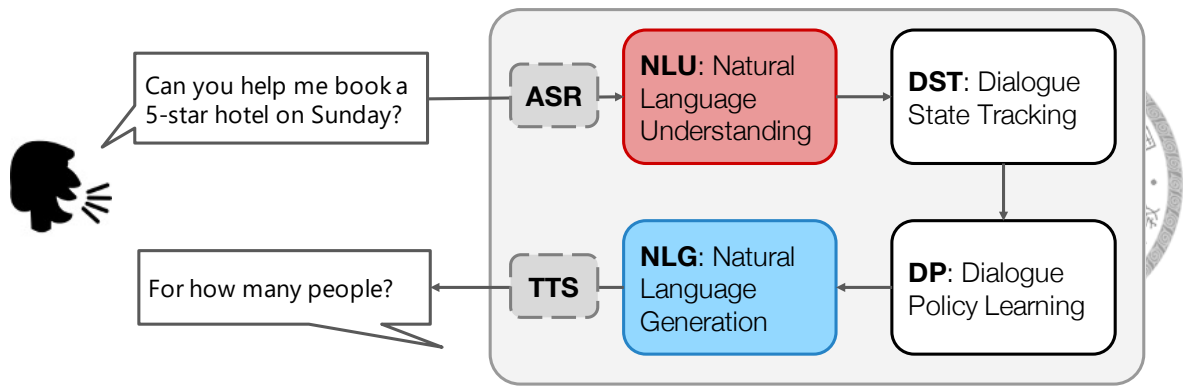


Figure 2.2: A typical dialogue system pipeline, which is composed of automatic speech recognition (ASR), natural language understanding (NLU), dialogue state tracking (DST), dialogue policy (DP), natural language generation (NLG), and Text-to-Speech (TTS) modules.

## 2.3 Dialogue Systems

Spoken dialogue systems that can help users solve complex tasks such as booking a movie ticket have become an emerging research topic in artificial intelligence and natural language processing areas. With a well-designed dialogue system as an intelligent personal assistant, people can accomplish certain tasks more easily via natural language interactions. Today, there are several virtual intelligent assistants on the market, such as Apple’s Siri, Google’s Home, Microsoft’s Cortana, and Amazon’s Echo. The recent advance of deep learning has inspired many applications of neural dialogue systems [8, 9, 10, 11]. A typical dialogue system pipeline can be divided into several components: an automatic speech recognizer (ASR) that transcribes a user’s speech input into texts, a natural language understanding module (NLU) to classify the domain along with domain-specific intents and fill in a set of slots to form a semantic frame [12, 13]. A dialogue state tracking (DST) module predicts the current dialogue state according to the multi-turn conversations, then the dialogue policy (DP) determines the system action for the next turn given the current dialogue state [14, 15]. Finally, the semantic frame indicating the policy is fed into a natural language generation (NLG) module to construct a response utterance to the user [16, 17, 18]. Finally, the generated utterances can be further transformed by a Text-to-Speech (TTS) module and then return to users. The pipeline is illustrated in Figure

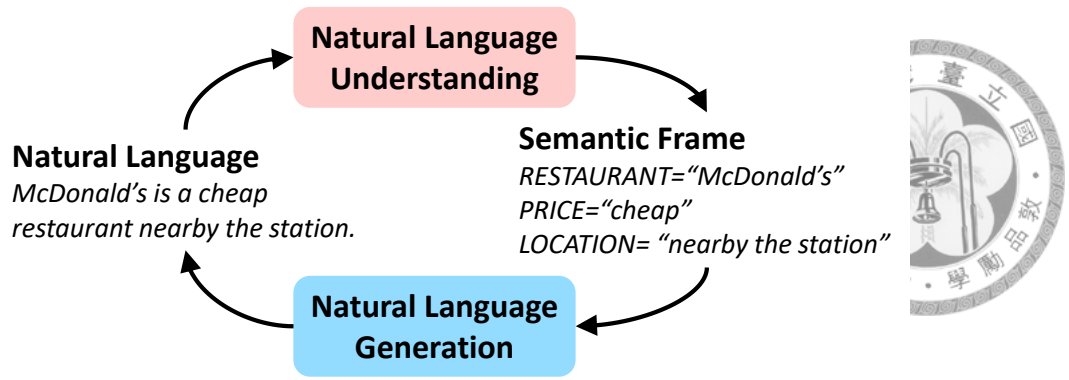


Figure 2.3: NLU and NLG emerge as a dual form.

2.2.

## 2.4 Problem Formulation

The problems we aim to solve are NLU and NLG; for both tasks, there are two spaces: the semantics space  $\mathcal{X}$  and the natural language space  $\mathcal{Y}$ . NLG is to generate sentences associated with the given semantics, where the goal is to learn a mapping function  $f : \mathcal{X} \rightarrow \mathcal{Y}$  that transforms semantic representations into natural language. On the other hand, NLU is to capture the core meaning of sentences, where the goal is to find a function  $g : \mathcal{Y} \rightarrow \mathcal{X}$  that predicts semantic representations from the given natural language. In other words, understanding and generating sentences are a dual problem pair shown in Figure 2.3.

Given  $n$  data pairs  $\{(x_i, y_i)\}_{i=1}^n$  *i.i.d.* sampled from the joint space  $\mathcal{X} \times \mathcal{Y}$ . A typical strategy for the optimization problem is based on maximum likelihood estimation (MLE) of the parameterized conditional distribution by the trainable parameters  $\theta_{x \rightarrow y}$  and  $\theta_{y \rightarrow x}$  as below:

$$f(x; \theta_{x \rightarrow y}) = \arg \max P(y \mid x; \theta_{x \rightarrow y}), \quad (2.10)$$

$$g(y; \theta_{y \rightarrow x}) = \arg \max P(x \mid y; \theta_{y \rightarrow x}), \quad (2.11)$$

where  $P(\cdot)$  is the estimated probability distribution, note that (2.10) and (2.11) means performing greedy decoding. Figure 2.4 and Figure 2.5 are simple examples of using recurrent models for NLU and NLG.

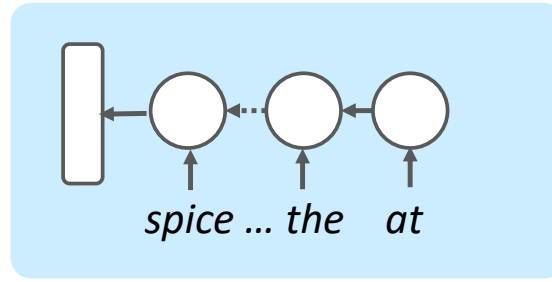


Figure 2.4: A simple example of using a recurrent unit as NLU model, which we encode an input sentence and use the final hidden state to predict semantic labels.

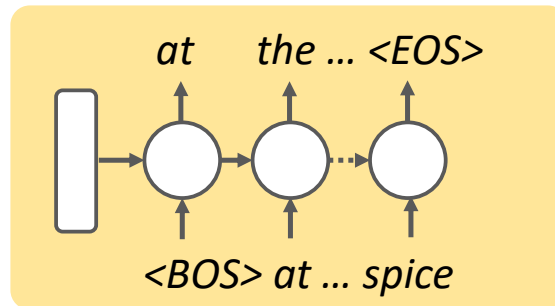


Figure 2.5: A simple example of using a recurrent unit as NLG model, which we feed a semantic vector and the begin-of-sentence (<BOS>) token into the model and predict a text token at each step.

## 2.5 Datasets

The benchmark datasets conducted in our experiments are listed as follows:

- **ATIS** [19]: an NLU dataset containing audio recordings of people making flight reservations. It has sentence-level intents and word-level slot tags.
- **SNIPS** [20]: an NLU dataset focusing on evaluating voice assistants for multiple domains, which has sentence-level intents and word-level slot tags.
- **E2E NLG** [21]: an NLG dataset in the restaurant domain, where each meaning representation has up to 5 references in natural language and no intent labels.

Since the above datasets are not designed for either NLU or NLG, we propose the following methods to augment the data.

Dataset	#Train	#Test	Vocab	#Intent	#Slot
SNIPS	13084	700	9076	7	72
ATIS	4478	893	1428	25	130
E2E NLG	42063	4693	3210	-	16



Table 2.1: The statistics of the datasets.

### 2.5.1 NLG $\rightarrow$ NLU

The E2E NLG challenge dataset [21]<sup>1</sup> is adopted in our experiments, which is a crowd-sourced dataset of 50k instances in the restaurant domain. Each instance is a pair of a semantic frame containing specific slots and corresponding values and a associated natural language utterance with the given semantics. For example, a semantic frame with the slot-value pairs “name[Bibimbap House], food[English], priceRange[moderate], area[riverside], near [Clare Hall]” corresponds to the target sentence “*Bibimbap House is a moderately priced restaurant who’s main cuisine is English food. You will find this local gem near Clare Hall in the Riverside area.*”. Although the original dataset is for NLG, of which the goal is to generate sentences based on the given slot-value pairs, we further formulate the NLU task as predicting slot-value pair based on the utterances, which can be viewed as a multi-label classification problem and each possible slot-value pair is treated as an individual label.

### 2.5.2 NLG $\rightarrow$ NLU (IOB)

The NLG dataset (E2E NLG) is augmented based on IOB schema and direct matching. For example, a semantic frame with the slot-value pairs:

{name[Bibimbap House], food[English],  
priceRange[moderate], area[riverside],  
near[Clare Hall]}

<sup>1</sup><http://www.macs.hw.ac.uk/InteractionLab/E2E/>

corresponds to the target sentence “*Bibimbap House is a moderately priced restaurant who’s main cuisine is English food. You will find this local gem near Clare Hall in the Riverside area.*”. The produced IOB slot data would be



[Bibimbap:B-Name, House:I-Name is:O a:O  
moderately:B-PriceRange, priced:I-PriceRange,  
restaurant:O, who’s:O, main:O, cuisine:O, is:O,  
English:B-Food food:O. You:O, will:O, find:O,  
this:O, local:O, gem:O, near:B-Near,  
Clare:I-Near, Hall:I-Near, in:O, the:O,  
Riverside:B-Area, area:I-Area].

### 2.5.3 NLU (IOB) → NLG

The modality of the NLU outputs (an intent and a sequence of IOB-slot tags) are different from the modality of the NLG inputs (semantic frame containing intent (if applicable) and slot-value pairs). Therefore, we propose a matching method: for each word, the NLU model will predict an IOB tag  $\in \{O, B\text{-slot}, I\text{-slot}\}$ , we simply drop the I- and B- and aggregate all the words with the same slot then combine it with the predicted intent.

For example, if given the word sequence:

[*which, flights, travel, from, kansas,*  
*city, to, los, angeles, on, april, ninth*],

the NLU predicts the IOB-slot sequence:

[O, O, O, O, B-fromloc.city\_name,  
I-fromloc.city\_name,  
O, B-toloc.city\_name, I-toloc.city\_name, O,  
B-depart\_date.month\_name,  
B-depart\_date.day\_number]



and a corresponding intent "atis\_flight", we transform the sequences into a semantic frame:

{intent[atis\_flight],  
fromloc.city\_name[kansas city],  
toloc.city\_name[los angelos],  
depart\_date.month\_name[april ninth]}.

The constructed semantic frames can then be used as the NLG input.





## Chapter 3

### Related Work

This paper focuses on modeling the duality between understanding and generation towards unsupervised learning of the two components, related work is summarized below.

#### 3.1 Natural Language Understanding

In dialogue systems, the first component is a natural language understanding (NLU) module— parsing user utterances into semantic frames that capture the core meaning [12]. A typical NLU first determines the domain given input utterances, predicts the intent, and then fill the associated slots [13, 22]. However, the above work focused on single-turn interactions, where each utterance is treated independently. To overcome the error propagation and further improve understanding performance, contextual information has been leveraged and shown useful [23, 24, 25, 26]. Also, different speaker roles provided informative signal for capturing speaking behaviors and achieving better understanding performance [27, 28].

#### 3.2 Natural Language Generation

NLG is another key component in dialogue systems, where the goal is to generate natural language sentences conditioned on the given semantics from the dialogue manager. As an endpoint of interacting with users, the quality of generated sentences is crucial for better user experience. In spite of robustness and adequacy of the rule-based methods, poor diver-

sity makes talking to a template-based machine unsatisfactory. Furthermore, scalability is an issue, because designing sophisticated rules for a specific domain is time-consuming. Previous work proposed a RNNLM-based NLG that can be trained on any corpus of dialogue act-utterance pairs without hand-crafted features and any semantic alignment [29]. The following work based on sequence-to-sequence (seq2seq) models further obtained better performance by employing encoder-decoder structure with linguistic knowledge such as syntax trees [30, 17, 18].

### 3.3 Dual Learning

Various tasks may have diverse goals, which are usually independent to each other. However, some tasks may hold a dual form, that is, we can swap the input and target of a task to formulate another task. Such structural duality emerges as one of the important relationship for further investigation. Two AI tasks are of structure duality if the goal of one task is to learn a function mapping from space  $\mathcal{X}$  to  $\mathcal{Y}$ , while the other's goal is to learn a reverse mapping from  $\mathcal{Y}$  and  $\mathcal{X}$ . Machine translation is an example [31], translation from English to Chinese has a dual task, which is translated from Chinese to English; the goal of automatic speech recognition (ASR) is opposite to the one of text-to-speech (TTS) [32], and so on. Previous work first exploited the duality of the task pairs and proposed supervised [33] and unsupervised (reinforcement learning) [34] learning frameworks. These recent studies magnified the importance of the duality by revealing exploitation of it could boost the learning of both tasks. [1] employed the dual supervised learning framework to train NLU and NLG and improve both models simultaneously. [35] improved models for conditional text generation using techniques from computational pragmatics. The techniques formulated language production as a game between speakers and listeners, where a speaker should generate text which a listener can use to correctly identify the original input the text describes. [36] proposed a semi-supervised framework to learn NLU with an auxiliary generation model for pseudo-labeling to make use of unlabeled data.



## Chapter 4

# Dual Supervised Learning

In this chapter, we propose a novel learning framework for natural language understanding and generation on top of standard supervised learning, which first exploits the duality between NLU and NLG and introduces it into the learning objective as the regularization term. The preliminary experiments show that the proposed approach boosts the performance for both tasks, demonstrating the effectiveness of the dual relationship.

### 4.1 Proposed Framework

This section first introduces the core training algorithm and then describe the proposed methods of estimating data distribution.

#### 4.1.1 Dual Supervised Learning

Considering the duality between two tasks in the dual problems, it is intuitive to bridge the bidirectional relationship from a probabilistic perspective. If the models of two tasks are optimal, we have *probabilistic duality*:

$$\begin{aligned} P(x)P(y \mid x; \theta_{x \rightarrow y}) &= P(y)P(x \mid y; \theta_{y \rightarrow x}) \\ &= P(x, y) \quad \forall x, y, \end{aligned}$$

where  $P(x)$  and  $P(y)$  are marginal distributions of data. The condition reflects parallel, bidirectional relationship between two tasks in the dual problem. Although standard supervised learning with respect to a given loss function is a straight-forward approach to address MLE, it does not consider the relationship between two tasks.

[33] exploited the duality of the dual problems to introduce a new learning scheme, which explicitly imposed the empirical probability duality on the objective function. The training strategy is based on the standard supervised learning and incorporates the probability duality constraint, so-called *dual supervised learning*. Therefore the training objective is extended to a multi-objective optimization problem:

$$\begin{cases} \min_{\theta_{x \rightarrow y}} (\mathbb{E}[l_1(f(x; \theta_{x \rightarrow y}), y)]), \\ \min_{\theta_{y \rightarrow x}} (\mathbb{E}[l_2(g(y; \theta_{y \rightarrow x}), x)]), \\ \text{s.t. } P(x)P(y | x; \theta_{x \rightarrow y}) = P(y)P(x | y; \theta_{y \rightarrow x}), \end{cases}$$

where  $l_{1,2}$  are the given loss functions. Such constraint optimization problem could be solved by introducing Lagrange multiplier to incorporate the constraint:

$$\begin{cases} \min_{\theta_{x \rightarrow y}} (\mathbb{E}[l_1(f(x; \theta_{x \rightarrow y}), y)] + \lambda_{x \rightarrow y} l_{duality}), \\ \min_{\theta_{y \rightarrow x}} (\mathbb{E}[l_2(g(y; \theta_{y \rightarrow x}), x)] + \lambda_{y \rightarrow x} l_{duality}), \end{cases}$$

where  $\lambda_{x \rightarrow y}$  and  $\lambda_{y \rightarrow x}$  are the Lagrange parameters and the constraint is formulated as follows:

$$\begin{aligned} l_{duality} = & (\log \hat{P}(x) + \log P(y | x; \theta_{x \rightarrow y}) \\ & - \log \hat{P}(y) - \log P(x | y; \theta_{y \rightarrow x}))^2. \end{aligned}$$

Now the entire objective could be viewed as the standard supervised learning with an additional regularization term considering the duality between tasks. Therefore, the learning scheme is to learn the models by minimizing the weighted combination of an original loss term and a regularization term. Note that the true marginal distribution of

data  $P(x)$  and  $P(y)$  are often intractable, so here we replace them with the approximated empirical marginal distribution  $\hat{P}(x)$  and  $\hat{P}(y)$ .



### 4.1.2 Distribution Estimation as Autoregression

With the above formulation, the current problem is how to estimate the empirical marginal distribution  $\hat{P}(\cdot)$ . To accurately estimate data distribution, the data properties should be considered, because different data types have different structural natures. For example, natural language has sequential structures and temporal dependencies, while other types of data may not. Therefore, we design a specific method of estimating distribution for each data type based on the expert knowledge.

From the probabilistic perspective, we can decompose any data distribution  $p(x)$  into the product of its nested conditional probability,

$$p(x) = \prod_d^D p(x_d \mid x_1, \dots, x_{d-1}), \quad (4.1)$$

where  $x$  could be any data type and  $d$  is the index of a variable unit.

### Language Modeling

Natural language has an intrinsic sequential nature; therefore it is intuitive to leverage the autoregressive property to learn a language model. In this work, we learn the language model based on recurrent neural networks [37, 38] by the cross entropy objective in an unsupervised manner.

$$p(y) = \prod_i^L p(y_i \mid y_1, \dots, y_{i-1}; \theta_y), \quad (4.2)$$

where  $y_{(\cdot)}$  are words in the sentence  $y$ , and  $L$  is the sentence length.

### Masked Autoencoder

The semantic representation  $x$  in our work is discrete semantic frames containing specific slots and corresponding values. Each semantic frame contains the core concept of a cer-

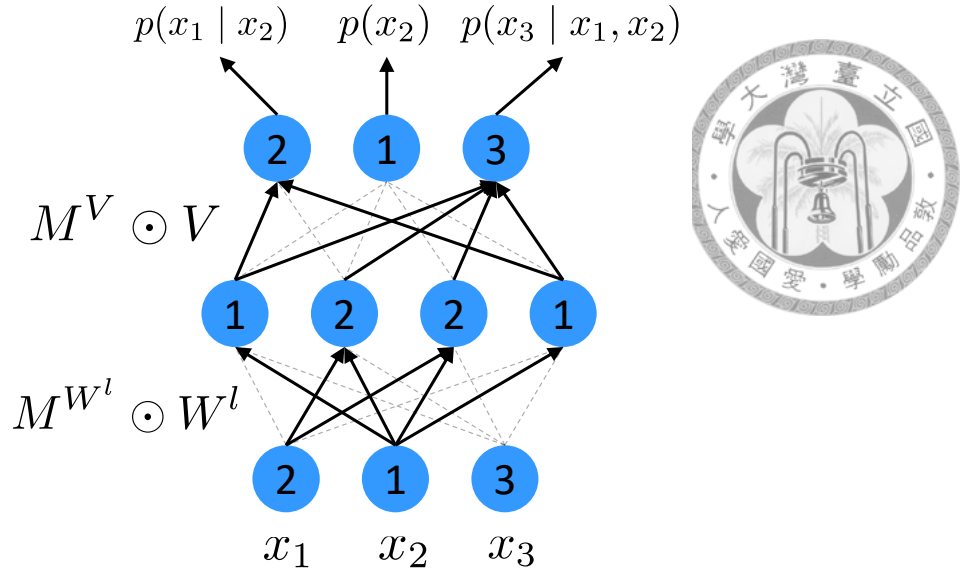


Figure 4.1: The illustration of the masked autoencoder for distribution estimation (MADE).

tain sentence, for example, the slot-value pairs “name[Bibimbap House], food[English], priceRange[moderate], area [riverside], near[Clare Hall]” corresponds to the target sentence “*Bibimbap House is a moderately priced restaurant who’s main cuisine is English food. You will find this local gem near Clare Hall in the Riverside area.*”. Even though the product rule in (4.1) enables us to decompose any probability distribution into a product of a sequence of conditional probability, how we decompose the distribution reflects a specific physical meaning. For example, language modeling outputs the probability distribution over vocabulary space of  $i$ -th word  $y_i$  by only taking the preceding word sequence  $y_{<i}$ . Natural language has the intrinsic sequential structure and temporal dependency, so modeling the joint distribution of words in a sequence by such autoregressive property is logically reasonable. However, slot-value pairs in semantic frames do not have a single directional relationship between them, while they parallel describe the same sentence, so treating a semantic frame as a sequence of slot-value pairs is not suitable. Furthermore, slot-value pairs are not independent, because the pairs in a semantic frame correspond to the same individual utterance. For example, French food would probably cost more. Therefore, the correlation should be taken into account when estimating the joint distribution.

Considering the above issues, to model the joint distribution of flat semantic frames,

various dependencies between slot-value semantics should be leveraged. In this work, we propose to utilize a masked autoencoder for distribution estimation (MADE) [39]. By zeroing certain connections, we could enforce the variable unit  $x_d$  to only depend on any specific set of variables, not necessary on  $x_{<d}$ ; eventually we could still have the marginal distribution by the product rule:

$$p(x) = \prod_d^D p(x_d | S_d), \quad (4.3)$$

where  $S_d$  is a specific set of variable units.

In practice, we elementwise-multiply each weight matrix by a binary mask matrix  $M$  to interrupt some connections, as illustrated in Figure 5.2. To impose the autoregressive property, we first assign each hidden unit  $k$  an integer  $m(k)$  ranging from 1 to the dimension of data  $D-1$  inclusively; for the input and output layers, we assign each unit a number ranging from 1 to  $D$  exclusively. Then binary mask matrices can be built as follows:

$$M = \begin{cases} 1 & \text{if } m^l(k') \geq m^{l-1}(k), \\ 1 & \text{if } m^L(d) > m^{L-1}(k), \\ 0 & \text{otherwise.} \end{cases}$$

Here  $l$  indicates the index of the hidden layer, and  $L$  indicates the one of the output layer. With the constructed mask matrices, the masked autoencoder is shown to be able to estimate the joint distribution as autoregression. Because there is no explicit rule specifying the exact dependencies between slot-value pairs in our data, we consider various dependencies by ensemble of multiple decomposition, that is, to sample different sets  $S_d$ .

## 4.2 Experiments

To evaluate the effectiveness of the proposed framework, we conduct the experiments, the settings and analysis of the results are described as follows.

Learning Scheme		NLU	NLG			
		F1	BLEU	R-1	R-2	R-L
(a)	Baseline: Iterative training	71.14	55.05	55.37	27.95	39.90
(b)	Dual supervised learning, $\lambda = 0.1$	<b>72.32</b>	<b>57.16</b>	<b>56.37</b>	<b>29.19</b>	<b>40.44</b>
(c)	Dual supervised learning, $\lambda = 0.01$	72.08	55.07	55.56	28.42	40.04
(d)	Dual supervised learning, $\lambda = 0.001$	71.71	56.17	55.90	28.44	40.08
(e)	Dual supervised learning w/o MADE	70.97	55.96	55.99	28.74	39.98

Table 4.1: The NLU performance reported on micro-F1 and the NLG performance reported on BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%).

### 4.2.1 Settings

The experiments are conducted in the benchmark E2E NLG challenge dataset [21], which is a crowd-sourced dataset of 50k instances in the restaurant domain. Our models are trained on the official training set and verified on the official testing set. Each instance is a pair of a semantic frame containing specific slots and corresponding values and an associated natural language utterance with the given semantics. The data preprocessing includes trimming punctuation marks, lemmatization, and turning all words into lowercase.

Although the original dataset is for NLG, of which the goal is to generate sentences based on the given slot-value pairs, we further formulate a NLU task as predicting slot-value pairs based on the utterances, which is a multi-label classification problem. Each possible slot-value pair is treated as an individual label, and the total number of labels is 79. The augmentation methods are described in Section 2.5.1. To evaluate the quality of the generated sequences regarding both precision and recall, for NLG, the evaluation metrics include BLEU and ROUGE (1, 2, L) scores with multiple references, while F1 score is measured for the NLU results.

### 4.2.2 Model Details

The model architectures for NLG and NLU are a gated recurrent unit (GRU) [5] with two identical fully-connected layers at the two ends of GRU, as illustrated in Figure 4.2 and Figure 4.2. Thus the model is symmetrical and may have semantic frame representation as initial and final hidden states and sentences as the sequential input.



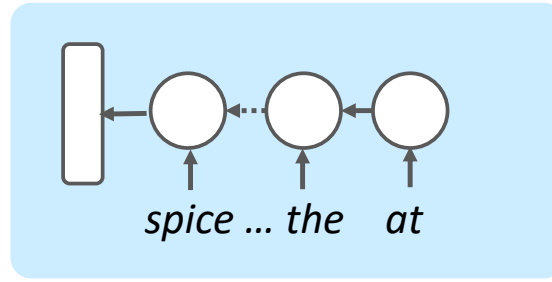


Figure 4.2: The NLU model, which we encode an input sentence and use the final hidden state to predict semantic labels.

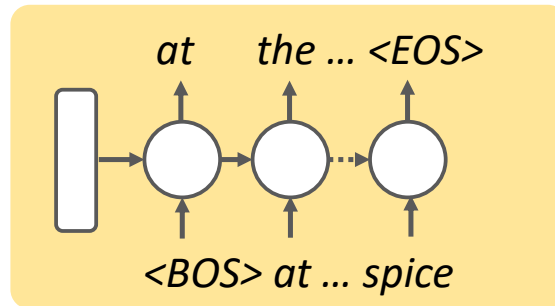


Figure 4.3: The NLG model, which we feed a semantic vector and the begin-of-sentence (<BOS>) token into the model and predict a text token at each step.

In all experiments, we use mini-batch *Adam* as the optimizer with each batch of 64 examples, 10 training epochs were performed without early stop, the hidden size of network layers is 200, and word embedding is of size 50 and trained in an end-to-end fashion.

### 4.2.3 Results and Analysis

The experimental results are shown in Table 4.1, where each reported number is averaged over three runs. The row (a) is the baseline that trains NLU and NLG separately and independently, and the rows (b)-(d) are the results from the proposed approach with different Lagrange parameters.

The proposed approach incorporates probability duality into the objective as the regularization term. To examine its effectiveness, we control the intensity of regularization by adjusting the Lagrange parameters. The results (rows (b)-(d)) show that the proposed method outperforms the baseline on all automatic evaluation metrics. Furthermore, the

performance improves more with stronger regularization (row (b)), demonstrating the importance of leveraging duality.

In this paper, we design the methods for estimating marginal distribution for data in NLG and NLU tasks: language modeling is utilized for sequential data (natural language utterances), while the masked autoencoder is conducted for flat representation (semantic frames). The proposed method for estimating the distribution of semantic frames considers complex and implicit dependencies between semantics by ensemble of multiple decomposition of joint distribution. In our experiments, the empirical marginal distribution is the average over the results from 10 different masks and orders; in other words, 10 types of dependencies are modeled. The row (e) can be viewed as the ablation test, where the marginal distribution of semantic frames is estimated by considering slot-value pairs independent to others and statistically computed from the training set. The performance is worse than the ones that model the dependencies, demonstrating the importance of considering the nature of input data and modeling data distribution via the masked autoencoder.

We further analyze understanding and generation results compared with the baseline model. In some cases, it is found that our NLU model can extract the semantics of utterances better and our NLG model can generate sentences with richer information based on the proposed learning scheme. In sum, the proposed approach is capable of improving the performance of both NLU and NLG in the benchmark data, where the exploitation of duality and the way of estimating distribution are demonstrated to be important.

### 4.3 Summary

This chapter proposes a novel training framework for natural language understanding and generation based on dual supervised learning, which first exploits the duality between NLU and NLG and introduces it into the learning objective as the regularization term. Moreover, expert knowledge is incorporated to design suitable approaches for estimating data distribution. The proposed methods demonstrate effectiveness by boosting the performance of both tasks simultaneously in the benchmark experiments.



## Chapter 5

# Joint Dual Learning

The prior chapter is the first attempt that utilized the duality between NLU and NLG to improve the performance via a dual supervised learning framework. However, the prior work still learned both components in a *supervised* manner; instead, this paper introduces a general learning framework to effectively exploit such duality, providing flexibility of incorporating both *supervised* and *unsupervised* learning algorithms to train language understanding and generation models in a joint fashion. The benchmark experiments demonstrate that the proposed approach is capable of boosting the performance of both NLU and NLG.

The contributions can be summarized as 3-fold:

- This work proposes a general learning framework using the duality between NLU and NLG, where *supervised* and *unsupervised* learning can be flexibly incorporated for joint training.
- This work is the first attempt to exploits the dual relationship between NLU and NLG towards unsupervised learning.
- The benchmark experiments demonstrate the effectiveness of the proposed framework.

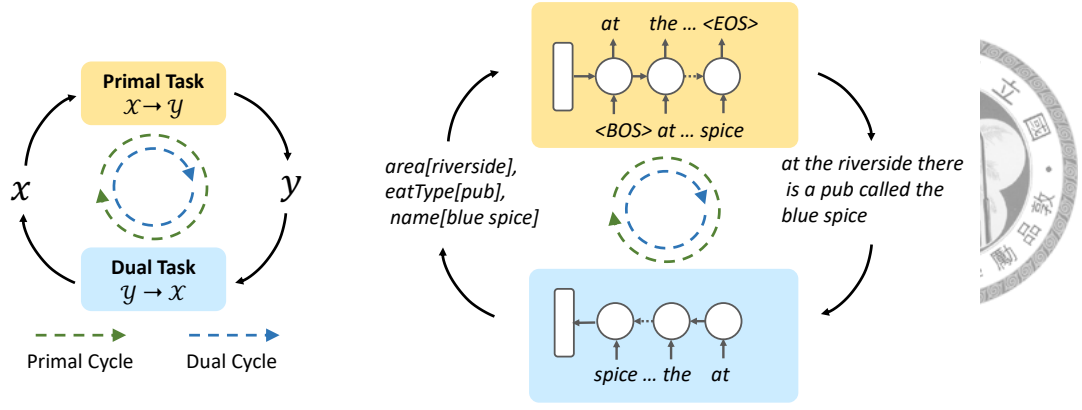


Figure 5.1: **Left:** The proposed joint dual learning framework, which comprises *Primal Cycle* and *Dual Cycle*. The framework is agnostic to learning objectives and the algorithm is detailed in Algorithm 1. **Right:** In our experiments, the models for NLG and NLU are a GRU unit accompanied with a fully-connected layer.

---

**Algorithm 1** Joint dual learning algorithm

---

- 1: **Input:** a mini-batch of  $n$  data pairs  $\{(x_i, y_i)\}_{i=1}^n$ , the function of the primal task  $f$ , the function of the dual task  $g$ , the loss function for the primal task  $l_1(\cdot)$ , the loss function for the dual task  $l_2(\cdot)$ , and the learning rates  $\gamma_1, \gamma_2$ ;
  - 2: **repeat**
  - 3:   Start from data  $x$ , transform  $x$  by function  $f$ :  $f(x_i; \theta_{x \rightarrow y})$ ; ▷ Primal Cycle
  - 4:   Compute the loss by  $l_1(\cdot)$ ;
  - 5:   Transform the output of the primal task by function  $g$ :  $g(f(x_i; \theta_{x \rightarrow y}); \theta_{y \rightarrow x})$ ;
  - 6:   Compute the loss by  $l_2(\cdot)$ ;
  - 7:   Update model parameters:
  - 8:    $\theta_{x \rightarrow y} \leftarrow \theta_{x \rightarrow y} - \gamma_1 \nabla_{\theta_{x \rightarrow y}} (\sum_{i=1}^n [l_1(f(x_i; \theta_{x \rightarrow y})) + l_2(g(f(x_i; \theta_{x \rightarrow y}); \theta_{y \rightarrow x}))])$ ;
  - 9:    $\theta_{y \rightarrow x} \leftarrow \theta_{y \rightarrow x} - \gamma_2 \nabla_{\theta_{y \rightarrow x}} (\sum_{i=1}^n [l_2(g(f(x_i; \theta_{x \rightarrow y}); \theta_{y \rightarrow x}))])$ ;
  - 10:   Start from data  $y$ , transform  $y$  by function  $g$ :  $g(y_i; \theta_{y \rightarrow x})$ ; ▷ Dual Cycle
  - 11:   Compute the loss by  $l_2(\cdot)$ ;
  - 12:   Transform the output of the dual task by function  $f$ :  $f(g(y_i; \theta_{y \rightarrow x}); \theta_{x \rightarrow y})$ ;
  - 13:   Compute the loss by  $l_1(\cdot)$ ;
  - 14:   Update model parameters:
  - 15:    $\theta_{y \rightarrow x} \leftarrow \theta_{y \rightarrow x} - \gamma_2 \nabla_{\theta_{y \rightarrow x}} (\sum_{i=1}^n [l_2(g(y_i; \theta_{y \rightarrow x})) + l_1(f(g(y_i; \theta_{y \rightarrow x}); \theta_{x \rightarrow y}))])$ ;
  - 16:    $\theta_{x \rightarrow y} \leftarrow \theta_{x \rightarrow y} - \gamma_1 \nabla_{\theta_{x \rightarrow y}} (\sum_{i=1}^n [l_1(f(g(y_i; \theta_{y \rightarrow x}); \theta_{x \rightarrow y}))])$ ;
  - 17: **until** convergence
- 

## 5.1 Proposed Framework

In this section, we describe the problem formulation and the proposed learning framework, which is illustrated in Figure 5.1 .

### 5.1.1 Joint Dual Learning

Although previous work has introduced the learning schemes that exploit duality of AI tasks, most of it was based on reinforcement learning or standard supervised learning and the models of primal and dual tasks ( $f$  and  $g$  respectively) are trained *separately*. Intuitively, if the models of primal and dual tasks are optimally learned, a complete cycle of transforming data from the original space to another space then back to the original space should be exactly the same as the original data, which could be viewed as the ultimate goal of a dual problem. In our scenario, if we generate sentences from given semantics  $x$  via the function  $f$  and transform them back to the original semantics perfectly via the function  $g$ , it implies that our generated sentences are grounded to the original given semantics and has the mathematical condition:

$$g(f(x)) \equiv x,$$

which is also known as *Cycle Consistency*. Therefore, our objective is to achieve the *perfect complete cycle* of data transforming by training two dual models ( $f$  and  $g$ ) in a *joint* manner.

#### Algorithm Description

As illustrated in Figure 5.1, the framework is composed of two parts: *Primal Cycle* and *Dual Cycle*. Primal Cycle starts from semantic frames  $x$ , (1) first transforms the semantic representation to sentences by the function  $f$ , (2) then computes the loss by the given loss function  $l_1$ , (3) predicts the semantic meaning from the generated sentences, (4) computes the loss by the given loss function  $l_2$ , (5) finally train the models based on the computed loss; Dual Cycle starts from utterances and is symmetrically formulated. The learning algorithm is described in Algorithm 1, which is agnostic to types of learning objective. Either a supervised learning objective or an unsupervised learning objective can be conducted at the end of the training cycles, and the whole framework can be trained in an *end-to-end* manner.

### 5.1.2 Learning Objective

As the language understanding task in our experiments is to predict corresponding slot-value pairs of utterances, which is a multi-label classification problem, we utilized the binary cross entropy loss as the supervised objective function for NLU. Likewise, the cross entropy loss function is used as the supervised objective for NLG. Take NLG for example, the objective of the model is to optimize the conditional probability of predicting word tokens given semantics  $p(y | x)$ , so that the difference between the predicted distribution and the target distribution,  $q(y | x)$ , can be minimized:

$$-\sum \sum_y^n q(y | x) \log p(y | x), \quad (5.1)$$

where  $n$  is the number of samples.

On the other hand, we can also introduce the reinforcement learning objective into our framework, the objective aims to maximize the expected value of accumulated reward. In our experiments, we conduct policy gradient (REINFORCE) method [40] for optimization, the gradient could be written as:

$$\nabla \mathbb{E}[r] = \mathbb{E}[r(y) \nabla \log p(y | x)], \quad (5.2)$$

where the variety of reward  $r$  will be elaborated in the next section. The loss function  $l_1$  for both tasks could be (5.1), (5.2), and the combination of them.

### 5.1.3 Reward Function

Different types of rewards reflect various objectives and would result in different behaviors in the learned policy. Hence, we design various reward functions to explore the model behavior, including explicit and implicit feedback.

#### Explicit Reward

To evaluate the quality of generated sentences, two explicit reward functions are adopted.

**Reconstruction Likelihood** In our scenario, if we generate sentences based on given semantics  $x$  by the function  $f$  and could transform them back to the original semantics perfectly by the function  $g$ , it implies our generated sentences ground on the original given semantics. Therefore we use the reconstruction likelihood at the end of the training cycles as a reward function:

$$\begin{cases} \log p(x \mid f(x_i; \theta_{x \rightarrow y}); \theta_{y \rightarrow x}) & \textbf{Primal}, \\ \log p(y \mid g(y_i; \theta_{y \rightarrow x}); \theta_{x \rightarrow y}) & \textbf{Dual}. \end{cases}$$

**Automatic Evaluation Score** The goal of most NLP tasks is to predict word tokens correctly, so the loss functions used to train these models focus on the word level, such as cross entropy maximizing the continuous probability distribution of the next correct word given the preceding context. However, the performance of these models is typically evaluated using discrete metrics. For instance, BLEU and ROUGE measure n-gram overlaps between the generated outputs and the reference texts. In order to enforce our NLG to generate better results in terms of the evaluation metrics, we utilize these automatic metrics as rewards to provide the sentence-level information. Moreover, we also leverage F-score in our NLU model to indicate the understanding performance.

### **Implicit Reward**

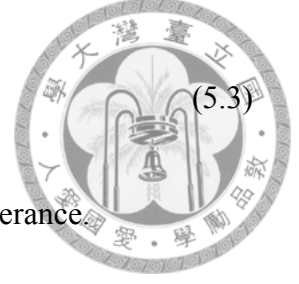
In addition to explicit signals like reconstruction likelihood and the automatic evaluation metrics, a “softer” feedback signal may be informative. For both tasks, we design model-based methods estimating data distribution in order to provide such soft feedback.

**Language Model** For NLG, we utilize pre-trained language models which estimate the whole data distribution to compute the joint probability of generated sentences, measuring their naturalness and fluency. In this work, we use a simple language model based on RNN [37, 38]. The language model is learned by a cross entropy objective in an unsuper-

vised manner:

$$p(y) = \prod_i^L p(y_i | y_1, \dots, y_{i-1}; \theta_y), \quad (5.3)$$

where  $y_{(\cdot)}$  are the words in a sentence  $y$ , and  $L$  is the length of the utterance



**Masked Autoencoder for Distribution Estimation (MADE)** For NLU, the output contains a set of discrete labels, which do not fit the sequential model scenarios such as language models. Each semantic frame  $x$  in our work contains the core concept of a certain sentence, furthermore, the slot-value pairs are not independent to others, because they correspond to the same individual utterance. For example, McDonald's would probably be inexpensive; therefore the correlation should be taken into account when estimating the joint distribution.

Following Chapter 4 [1], we measure the soft feedback signal for NLU using masked autoencoder [39] to estimate the joint distribution. By interrupting certain connections between hidden layers, we could enforce the variable unit  $x_d$  to only depend on any specific set of variables, not necessary on  $x_{<d}$ ; eventually we could still have the joint distribution by product rule:

$$p(x) = \prod_d^D p(x_d | S_d),$$

where  $d$  is the index of variable unit,  $D$  is the total number of variables, and  $S_d$  is a specific set of variable units. Because there is no explicit rule specifying the exact dependencies between slot-value pairs in our data, we consider various dependencies by ensembles of multiple decomposition by sampling different sets  $S_d$  and averaging the results.

#### 5.1.4 Flexibility of Learning Scheme

The proposed framework provides various flexibility of designing and extending the learning scheme, described as follows.

**Straight-Through Estimator** In many NLP tasks, the learning targets are discrete, so the goals of most NLP tasks are predicting discrete labels such as words. In practice we



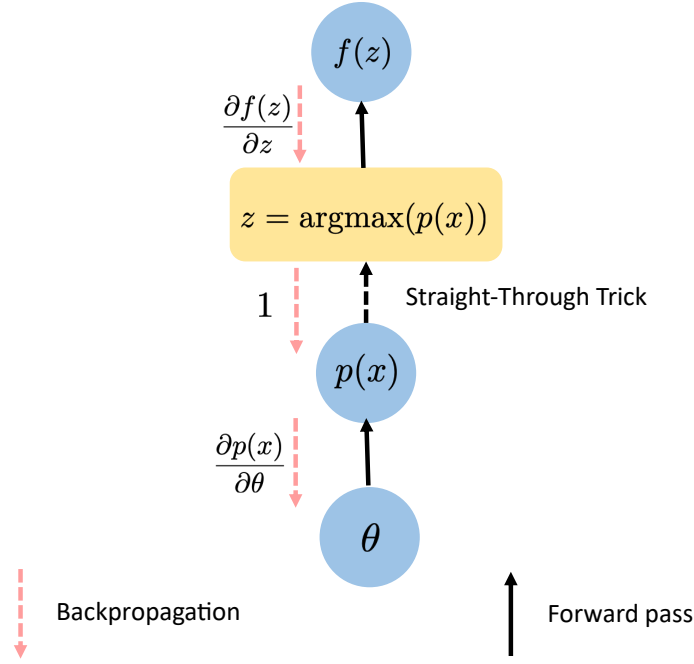


Figure 5.2: Straight-Through estimator.

perform  $\operatorname{argmax}$  operations on the output distribution from learned models to select the most possible candidates. However, such operation does not have any gradient value, forbidding the networks be trained via backpropagation. Therefore, it is difficult to directly connect a primal task (NLU in our scenario) and a dual task (NLG in our scenario) and jointly train these two models due to the above issue.

The Straight-Through (ST) estimator [41] is a widely applied method due to its simplicity and effectiveness. The idea of Straight-Through estimator is directly using the gradients of discrete samples as the gradients of the distribution parameters. Because discrete samples could be generated as the output of hard threshold functions or some operations on the continuous distribution, [41] explained the estimator by setting the gradients of hard threshold functions to 1. The structure of the Straight-Through estimator is illustrated in Figure 5.2. In this work, we introduce ST estimator for connecting two models, and therefore the gradient can be estimated and two models can be jointly trained in an end-to-end manner.

**Distribution as Input** In addition to employing the Straight-Through estimator, an alternative solution is to use continuous distribution as the input of models. For NLU, the

inputs are the word tokens from NLG, so we use the predicted distribution over the vocabulary to perform the weighted-sum of word embeddings. For NLG, the model requires semantic frame vectors predicted by NLU as the input condition; in this case, the probability distribution of slot-value pairs predicted by NLU can directly serve as the input vector. By utilizing the output distribution in this way, two models can be trained jointly in an end-to-end fashion.

**Hybrid Objective** As described before, the proposed approach is agnostic to learning algorithms; in other words, we could apply different learning algorithms at the middle and end of the cycles. For example, we could apply supervised learning on NLU in the first half of Primal Cycle and reinforcement learning on NLG to form a hybrid training cycle. Because two models are trained jointly, the objective applied on one model would potentially impact on the behavior of the other. Furthermore, we could also apply multiple objective functions including supervised or unsupervised ones to formulate multi-objective learning schemes.

**Towards Unsupervised Learning** Because the whole framework can be trained jointly and propagate the gradients, we could apply only one objective in one learning cycle at the end of it. Specifically, in Algorithm 1, we can apply only  $l_2$  in line 8 and only  $l_1$  in line 15. Such flexibility potentially enables us to train the models based on unpaired data in a unsupervised manner. For example, sample unpaired data  $x$  and transform the data by function  $f$ , next, feed them into the function  $g$ , then compare the predicted results and the original input to compute the loss. Likewise, we can perform the training cycle symmetrically from  $y$ . It is also possible to utilize limited data and perform the autoencoding cycle described above to apply semi-supervised learning.

## 5.2 Experiments

Our models are trained on the official training set and verified on the official testing set of the E2E NLG challenge dataset [21]. The data preprocessing includes trimming punctua-

Learning Scheme		NLU	NLG			
		Micro-F	BLEU	R-1	R-2	R-L
(a)	Iterative Training ( <b>Supervised</b> )	71.14	55.05	55.37	27.95	39.90
(b)	Dual Supervised Learning [1]	<b>72.32</b>	<b>57.16</b>	<b>56.37</b>	<b>29.19</b>	<b>40.44</b>
(c)	Joint Training (Straight-Through)	71.73	55.19	55.16	27.45	39.33
(d)	(c) + (NLG w/ distribution)	73.22	55.18	55.35	27.81	<b>39.36</b>
(e)	(c) + (NLU w/ distribution)	79.19	51.47	53.62	26.17	37.90
(f)	(c) + (NLU and NLG w/ distribution)	<b>80.03</b>	<b>55.34</b>	<b>56.17</b>	<b>28.48</b>	39.24
(g)	(f) + $\mathbf{RL}_{mid}$ (reconstruction likelihood)	80.07	55.32	56.12	28.07	39.59
(h)	(f) + $\mathbf{RL}_{end}$ (reconstruction likelihood)	79.97	55.21	56.15	28.50	39.42
(i)	(f) + $\mathbf{RL}_{mid}$ (BLEU+ROUGE, F1)	79.49	56.04	56.61	28.78	39.93
(j)	(f) + $\mathbf{RL}_{end}$ (BLEU+ROUGE, F1)	80.35	<b>57.59</b>	<b>56.71</b>	<b>29.06</b>	<b>40.28</b>
(k)	(f) + $\mathbf{RL}_{mid}$ (LM, MADE)	<b>81.52</b>	54.13	54.60	26.85	38.90
(l)	(f) + $\mathbf{RL}_{end}$ (LM, MADE)	79.52	55.61	55.97	28.57	39.97

Table 5.1: The NLU performance reported on micro-F and the NLG performance reported on BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%).

tion marks, lemmatization, and turning all words into lowercase. Each possible slot-value pair is treated as an individual label and the total number of labels is 79. The augmentation methods are described in Section 2.5.1. To evaluate the quality of the generated sequences regarding both precision and recall, for NLG, the evaluation metrics include BLEU and ROUGE (1, 2, L) scores with multiple references, while F1 measure is reported for evaluating NLU.

### 5.2.1 Model

The proposed framework and algorithm are agnostic to model structures. In our experiments, we use a gated recurrent unit (GRU) [5] with fully-connected layers at ends of GRU for both NLU and NLG, which are illustrated in the right part of Figure 6.1. Thus the models may have semantic frame representation as initial and final hidden states and sentences as the sequential input. In all experiments, we use mini-batch *Adam* as the optimizer with each batch of 64 examples. 10 training epochs were performed without early stop, the hidden size of network layers is 200, and word embedding is of size 50.

## 5.2.2 Results and Analysis

The experimental results are shown in Table 5.1, each reported number is averaged on the official testing set from three turns. Row (a) is the baseline where NLU and NLG models are trained independently and separately by supervised learning. The best performance in [1] is reported in row (b), where NLU and NLG are trained separately by supervised learning with regularization terms exploiting the duality.

To overcome the issue of non-differentiability, we introduce Straight-Through estimator when connecting two tasks. Based on our framework, another baseline for comparison is to train two models jointly by supervised loss and straight-through estimators, of which the performance is reported in row (c). Specifically, the cross entropy loss (5.1) is utilized in both  $l_1$  and  $l_2$  in Algorithm 1. Because the models in the proposed framework are trained jointly, the gradients are able to flow through the whole network thus two models would directly influence learning of each other. Rows (d)-(f) show the ablation experiments for exploring the interaction between two models ( $f$  and  $g$ ). For instance, row (e) does not use ST at the output of the NLU module; instead, we feed continuous distribution over slot-value labels instead of discrete semantic frames into NLG as the input. Instead of discrete word labels, row (d) and row (f) feed weighted sum over word embeddings based on output distributions. Since the goal of NLU is to learn a many-to-one function, considering all possibility would potentially benefit learning (row (d)-(f)).

On the contrary, the goal of NLG is to learn a one-to-many function, applying the ST estimator at the output of NLU only rather than both sides degrades the performance of generation (row (e)). However, this model achieves unexpected improvement in understanding by over 10%, the reason may be the following. The semantics representation is very compact, a slight noise in the semantics space would possibly result in a large difference in the target space and a totally different semantic meaning. Hence the continuous distribution over slot-value pairs may potentially cover the unseen mixture of semantics and further provide rich gradient signals. This could also be explained from the perspective of data augmentation. Moreover, connecting two models with continuous distribution at both joints further achieves improvement in both NLU and NLG (row (f)). Although

	Baseline	Proposed
$x$	area[riverside], eatType[pub], name[blue spice]	
$y$	<i>at the riverside there is a pub called the blue spice</i>	
$f(x; \theta_{x \rightarrow y})$	<i>blue spice is a pub in riverside that has a price range of more than 30e</i>	<i>in riverside there is a pub called blue spice</i>
$g(f(x; \theta_{x \rightarrow y}); \theta_{y \rightarrow x})$	area[city centre], customer rating[5 out of 5], priceRange[more than 30], priceRange[cheap], name[blue spice], name[the vaults]	area[riverside], eatType[pub], name[blue spice]

Table 5.2: An example of the Primal Cycle, where the baseline model is row (a) in Table 5.1.

	Baseline	Proposed
$y$	<i>blue spice is a family friendly pub located in the city centre it serves chinese food and is near the rainbow vegetarian cafe</i>	
$x$	familyFriendly[yes], area[city centre], eatType[pub], food[chinese], name[blue spice], near[rainbow vegetarian cafe]	
$g(y; \theta_{y \rightarrow x})$	familyFriendly[yes], food:[chinese]	familyFriendly[yes], area[city centre], eatType[pub], priceRange[moderate], food[chinese], name[blue spice]
$f(g(y; \theta_{y \rightarrow x}); \theta_{x \rightarrow y})$	<i>the chinese restaurant the twenty two is a family friendly restaurant</i>	<i>the chinese restaurant the blue spice is located in the city centre it is moderately priced and kid friendly</i>

Table 5.3: An example of the Dual Cycle, where the baseline model is row (a) in Table 5.1.

row (f) performs best in our experiments and dataset, as most AI tasks are classification problems, the proposed framework with ST estimators provides a general way to connect two tasks with duality. The proposed methods also significantly outperform the previously proposed dual supervised learning framework [1] on F1 score of NLU and BLEU score of NLG, demonstrating the benefit of learning NLU and NLG jointly.

### 5.2.3 Investigation of Hybrid Objectives

The proposed framework provides the flexibility of applying multiple objectives and different types of learning methods. In our experiments, apart from training two models jointly by supervised loss, reinforcement learning objectives are also incorporated into the training schemes (row (g)-(l)). The ultimate goal of reinforcement learning is to maximize the expected reward in (5.2). In the proposed dual framework, if we take expectation

over different distribution, it would reflect a different physical meaning. For instance, if we receive a reward at the end of Primal Cycle and the expectation is taken over the output distribution of NLG (middle) or NLU (end), the derivatives of objective functions would differ:

$$\begin{cases} \mathbb{E}[r_i \nabla \log p(y_i | x; \theta_{x \rightarrow y})] & \mathbf{RL}_{mid}, \\ \mathbb{E}[r_i \nabla \log p(x_i | f(x; \theta_{x \rightarrow y}); \theta_{y \rightarrow x})] & \mathbf{RL}_{end}. \end{cases}$$

The upper one ( $\mathbf{RL}_{mid}$ ) assesses the expected reward earned by the sentences constructed by the policy of NLG, which is a direct signal for the primal task NLG. The lower one ( $\mathbf{RL}_{end}$ ) estimates the expected reward earned by the predicted semantics by the policy of NLU based on the state predicted by NLG, such reward is another type of feedback.

In the proposed framework, the models of two tasks are trained jointly, thus an objective function will simultaneously influence the learning of both models. Different reward designs could guide reinforcement learning agents to different behaviors. To explore the impact of reinforcement learning signal, various rewards are applied on top of the joint framework (row (f)):

1. Token-level likelihood (rows (g) and (h)),
2. Sentence/frame-level automatic evaluation metrics (rows (i) and (j)),
3. Corpus-level joint distribution estimation (rows (k) and (l)).

In other words, the models in rows (g)-(l) have both supervised and reinforcement learning signal. The results show that token-level feedback may not provide extra guidance (rows (g) and (h)), directly optimizing towards the evaluation metrics at the testing phase benefits learning in both tasks and performs best (rows (i) and (j)), and the models utilizing learning-based joint distribution estimation also obtain improvement (row (k)). In sum, the explicit feedback is more useful for boosting the NLG performance, because the reconstruction and automatic scores directly reflect the generation quality. However, the implicit feedback is more informative for improving NLU, where MADE captures the salient information for building better NLU models. The results align well with the finding in [1].

## 5.2.4 Qualitative Analysis

Table 5.2 and 5.3 show the selected examples of the proposed model and the baseline model in Primal and Dual Cycle. As depicted in Algorithm 1, Primal Cycle is designed to start from semantic frames  $x$ , then transform the representation by the NLG model  $f$ , finally feed the generated sentences into the NLU model  $g$  and compare the results with the original input to compute loss. In the example of Primal Cycle (Table 5.2), we can find that  $f(g(y; \theta_{y \rightarrow x}); \theta_{x \rightarrow y})$  equals  $x$ , which means the proposed method can successfully restore the original semantics. On the other hand, Dual Cycle starts from natural language utterances, from the generated results (Table 5.3) we can find that our proposed method would not lose semantic concepts in the middle of the training cycle  $(g(y; \theta_{y \rightarrow x})) \leftrightarrow x$ . Based on the qualitative analysis, we can find that by considering the duality into the objective and jointly training, the proposed framework can improve the performance of NLU and NLG simultaneously.

## 5.3 Summary

This chapter proposes a general learning framework leveraging the duality between language understanding and generation, providing the flexibility of incorporating supervised and unsupervised learning algorithms to jointly train two models. The proposed framework provides a potential method towards unsupervised learning of both language understanding and generation models by considering their data distribution. The experiments on the benchmark dataset demonstrate that the proposed approach is capable of boosting the performance of both NLU and NLG models, motivating the potential research directions in this area.







## Chapter 6

# Dual Mutual Information Maximization

In previous chapters, we have faced some challenges during the experimental process, for example, NLG is much harder to optimize than NLU. We presume that the reasons are (1) the natural language space and the semantic space are quite different, and (2) the asymmetric relationship between NLU (many-to-one) and NLG (one-to-many). Hence in this chapter, we aim to enhance the joint learning framework by maximizing the mutual information between the representation of language and semantics in the hidden spaces.

## 6.1 Proposed Method

This section describes the problem formulation and the proposed learning framework illustrated in Figure 6.1.

### 6.1.1 Conversations as Transmission of Semantics

During conversations, the goal of communication is to understand what your interlocutors want to convey, which is similar to the goal of a channel defined in information theory for digital communication [42]. Specifically, when transmitting signals, the goal is to reduce the noise during transmission and hence successful reconstruction. Therefore, the process of how human converses is defined as the communication of thoughts, which contains two phases: (1) converting one's thoughts into messages (NLG), and (2) trying to understand the received messages (NLU). The characteristics of a channel such as a rate distortion

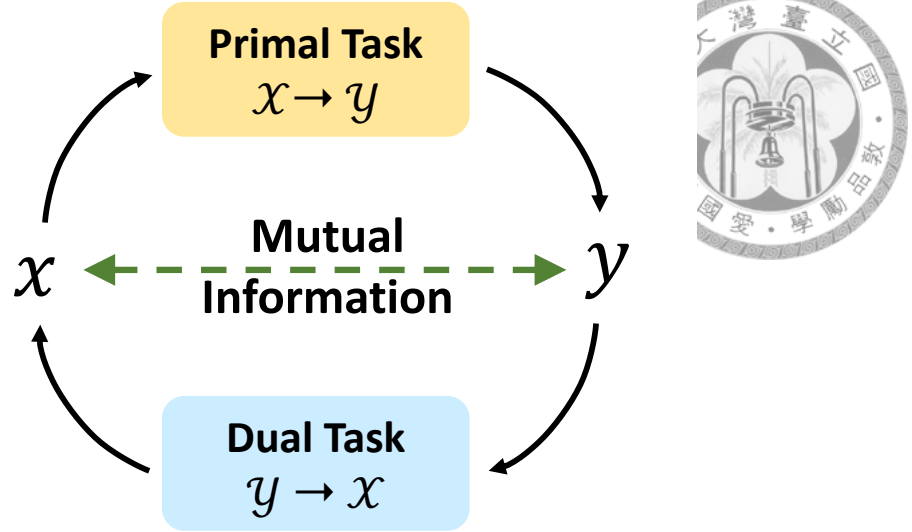


Figure 6.1: The proposed framework, which we aim to improve the mutual information between the representation of language and semantics.

function and the channel capacity depend on the mutual information between input signals and output signals. The channel capacity is defined as the maximum of the mutual information between the input and output of the channel, which is the tight upper bound for the rate at which information can be reliably transmitted over a communication channel. One of core concepts in dual learning algorithms proposed in the previous chapters is to leverage the reconstruction property that if a data point goes through two models in a dual task pair successively, it can be well reconstructed. Therefore, conversations are modeled as a channel by a chain of NLG and NLU, and we aim at maximizing the mutual information between input and output, which is analogous to have a larger channel capacity so that the transmitted information could be properly reconstructed (Figure 6.3).

### 6.1.2 Mutual Information Estimation

The mutual information (MI) is equivalent to the Kullback-Leibler (KL-) divergence between the joint distribution and the product of marginals of two random variables. MI cannot be directly used as a training objective since being intractable, so the previous

### Natural Language

1. Alimentum city centre is family-friendly.
2. Alimentum is a family-friendly city centre.

### Semantic Frame

NAME="Alimentum"  
familyFriendly="yes"  
area="city centre"

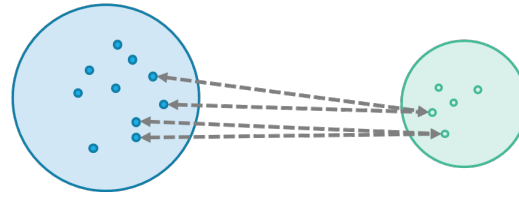


Figure 6.2: The illustration of projections between the natural language space and the semantic space.

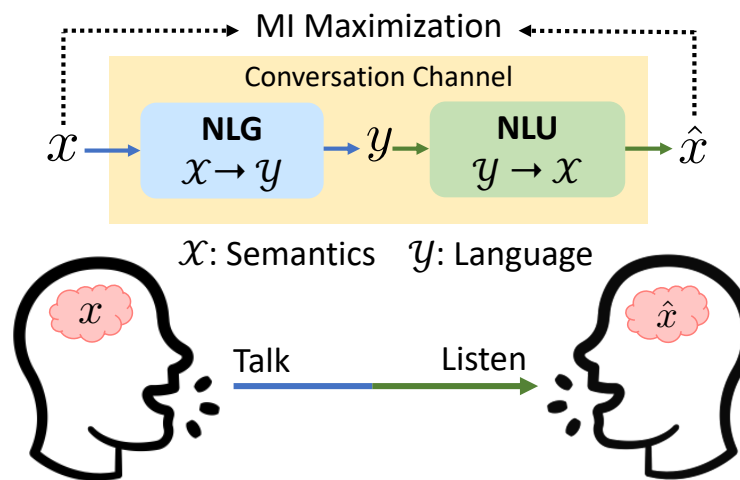


Figure 6.3: The proposed framework and an analogy of how human converses for one-way transmission of the semantics  $x$ .

work often utilized MI only in the inference stage to select a best candidate from a pool [43].

Recently, MINE [44] and Deep Infomax (DIM) [45] enabled estimating MI by back-propagation in neural networks. They estimated mutual information by training a discriminator to distinguish between positive samples from the joint distribution and negative samples from the product of marginals. DIM proposes to use Jensen-Shannon divergence, which can be efficiently implemented using the binary cross-entropy (BCE) loss. In our

experiments, we follow the DIM variant proposed in [46]:

$$\begin{aligned}
 MI(X; Y) \geq & \mathbb{E}_{\mathbb{P}}[\log(d(x, y))] + \\
 & \frac{1}{2} \mathbb{E}_{\mathbb{N}}[\log(1 - d(x, \bar{y}))] + \\
 & \frac{1}{2} \mathbb{E}_{\mathbb{N}}[\log(1 - d(\bar{x}, y))],
 \end{aligned} \tag{6.1}$$



where  $d(\cdot)$  is a learned discriminator function, which can be parameterized by neural networks,  $\mathbb{E}_{\mathbb{P}}$  and  $\mathbb{E}_{\mathbb{N}}$  denote the expectation over positive and negative samples respectively, and  $(\bar{x}, y)$  and  $(x, \bar{y})$  are negative samples from the product of marginals.

### 6.1.3 Training

Inspired by [2], we propose a back-and-forth learning scheme, which contains (1) starting from semantic representations and going through NLG and NLU in a row, and (2) starting from word sequences and going through NLU and NLG successively. Besides standard MLE objectives, (6.1) is introduced as a regularizer. For simplicity, we propose a batch learning algorithm detailed in Algorithm 2.

The core concept of the algorithm is to encourage the models to maximize the mutual information between semantic representations and language representations during training. Different modalities of representations can be used in (6.1); for example, we utilize semantic frame encoding as  $x$  and the distribution over words as  $y$  in our implementation.

## 6.2 Experiments

This work follows the commonly used slot filling and intent prediction setting [47], where the NLU benchmarks ATIS [19] and SNIPS [20] are adopted. To fairly examine the effectiveness of the proposed method, we keep the model structure simple by using the GRU-based models for both NLU and NLG. The model details can be found in Appendix . For NLG, the evaluation metrics include BLEU and ROUGE-(1, 2, L) scores with multiple references, while for NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The benchmark datasets conducted in our experiments are

---

**Algorithm 2** Batch learning algorithm

---

- 1: **Input:** the function of NLU  $f$ , the function of NLG  $g$ , the standard MLE objectives  $\mathcal{L}_f(\cdot)$  and  $\mathcal{L}_g(\cdot)$  and the learning rate  $\gamma$  and the weight  $\lambda$  for MI regularization  $\mathcal{L}_{MI}(\cdot)$ ;
  - 2: **repeat**
  - 3:     Sample a mini-batch  $B$  with  $n$  data pairs  $(x, y)$ ;
  - 4:     **NLG  $\rightarrow$  NLU**
  - 5:     Start from semantics  $x$ , transform  $x$  by NLG function  $f$ :  $\hat{y} = f(x; \theta_{x \rightarrow y})$ ;
  - 6:     Compute the loss  $\mathcal{L}_f(\hat{y}, y)$ ;
  - 7:     Random shuffle  $B$  and map the data pairs to original order to have negative samples  $(\bar{x}, \hat{y})$  and  $(x, \bar{y})$  ;
  - 8:     Compute MI regularization (Equation (6.1)):  
        $\mathcal{L}_{MI} = \frac{1}{n} \sum \log(d(x, \hat{y})) + \log(1 - d(x, \bar{y})) + \log(1 - d(\bar{x}, \hat{y}))$ ;
  - 9:     Transform  $\hat{y}$  by NLU function  $g$ :  $\hat{x} = g(\hat{y}; \theta_{y \rightarrow x})$
  - 10:    Compute the loss  $\mathcal{L}_g(\hat{x}, x)$ ;
  - 11:    Random shuffle  $B$  and map the data pairs to original order to have negative samples  $(\hat{x}, \bar{y})$  and  $(\bar{x}, y)$  ;
  - 12:    Compute MI regularization (Equation (6.1)):  
        $\mathcal{L}_{MI} = \frac{1}{n} \sum \log(d(\hat{x}, y)) + \log(1 - d(\bar{x}, y)) + \log(1 - d(\hat{x}, \bar{y}))$ ;
  - 13:    Transform  $\hat{x}$  by NLG function  $f$ :  $\hat{y} = f(\hat{x}; \theta_{x \rightarrow y})$
  - 14:    Compute the loss  $\mathcal{L}_f(\hat{y}, y)$ ;
  - 15:    // Update model parameters by the computed loss:
  - 16:     $\theta_{x \rightarrow y} \leftarrow \theta_{x \rightarrow y} - \gamma \nabla_{\theta_{x \rightarrow y}} (\sum \mathcal{L}_f(\cdot) + \sum \mathcal{L}_g(\cdot) - \lambda \sum \mathcal{L}_{MI}(\cdot))$ ;
  - 17:     $\theta_{y \rightarrow x} \leftarrow \theta_{y \rightarrow x} - \gamma \nabla_{\theta_{y \rightarrow x}} (\sum \mathcal{L}_f(\cdot) + \sum \mathcal{L}_g(\cdot) - \lambda \sum \mathcal{L}_{MI}(\cdot))$ ;
  - 18: **until** convergence
- 

ATIS [19], SNIPS [20], and E2E NLG [21]. The NLG datasets are augmented into IOB-format NLU data, the augmentation methods are described in Section 2.5.2 and 2.5.3. The hyperparameters and training details are reported in Section 7.2.1.

### 6.2.1 Training Details

In all the experiments, we use mini-batch Adam as the optimizer with each batch of 48 examples on Nvidia Tesla V100. 10 training epochs were performed without early stop, the hidden size of network layers is 200, and word embedding is of size 50. The ratio of teacher forcing is set to 0.9, the weights of mutual information regularization ( $\lambda$ ) are different in different models and selected by grid search in  $(0, 1]$  with step 0.1.

Learning Scheme		NLU		NLG			
		Accuracy	F1	BLEU	R-1	R-2	R-L
<b>ATIS</b>							
(a)	Iterative Baseline	85.98	<b>96.28</b>	16.71	37.11	13.47	35.88
(b)	Dual Supervised Learning	83.02	94.73	16.72	37.89	14.60	36.53
(c)	Joint Baseline	80.61	91.26	17.26	38.10	14.69	36.73
(d)	(c) + MI(semantics, word)	88.15	93.75	<b>24.46</b>	<b>42.92</b>	<b>23.01</b>	<b>41.78</b>
(e)	(c) + MI(semantics, sentence)	<b>88.50</b>	93.85	19.28	39.55	16.88	38.19
<b>SNIPS</b>							
(f)	Iterative Baseline	<b>97.40</b>	<b>96.98</b>	14.69	35.2	13.27	34.19
(g)	Dual Supervised Learning	97.39	96.35	15.90	39.85	16.39	38.69
(h)	Joint Baseline	97.32	94.56	17.19	38.59	16.36	37.53
(i)	(h) + MI(semantics, word)	97.02	94.25	<b>19.30</b>	<b>42.20</b>	<b>19.66</b>	<b>40.83</b>
(j)	(h) + MI(semantics, sentence)	96.93	95.42	16.82	39.06	16.45	37.75
<b>E2E NLG</b>							
(k)	Iterative Baseline	-	<b>94.41</b>	18.21	31.66	12.47	27.39
(l)	Dual Supervised Learning	-	94.36	24.32	45.91	19.31	39.92
(m)	Joint Baseline	-	92.69	24.47	45.41	19.22	39.10
(n)	(m) + MI(semantics, word)	-	92.69	<b>40.53</b>	<b>61.00</b>	<b>36.14</b>	<b>52.60</b>
(o)	(m) + MI(semantics, sentence)	-	92.64	28.21	49.52	23.18	41.63

Table 6.1: For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). All reported numbers are averaged over three different runs.

### 6.2.2 Model Structure

To evaluate the proposed methods on a fair basis, we make the NN model structures simple. For NLU, the model is a simple GRU [5] with a word and last output as input at each timestep  $i$  and a linear layer at the end for predicting intent based on the final hidden state:

$$o_i = \text{GRU}([w_i, o_{i-1}]).$$

The model for NLG is almost the same but with an additional encoder for encoding semantic frames, where slot-value pairs are encoded into semantic vectors for basic attention, the mean-pooled semantic vector is used as initial state. We borrow the encoder structure in [36] in our experiments. At each timestep  $i$ , the last predicted word and the

aggregated semantic vector from attention are used as the input:

$$o_i = \text{GRU}([h_i^{\text{Attn}}, o_{i-1}] \mid h_{\text{mean}}).$$

For the discriminator  $d$  in DIM objective (6.1), we follow [46] to use a bilinear function:

$$d(x, y) = x^T W y, \quad (6.2)$$

where  $W$  is a trainable weight matrix. Note that in our experiments, we use the mean-pooled semantic representation  $h_{\text{mean}}$  as  $x$  in (6.2).

### 6.2.3 Results and Analysis

The results are shown in Table 6.1, where there are three baselines for each dataset: (1) Iterative Baseline: simply training NLU and NLG iteratively, (2) Dual Supervised Learning [1], and (3) Joint Baseline: the output from one model is sent to another as in [2]. However, in our NLU setting, it is infeasible to flow the gradients through the loop for training the models jointly. In joint baselines, the outputs of NLU are the intent and IOB-slot tags, whose modalities are different from the NLG input, so the matching method in 2.5.3 is performed.

We apply the proposed method to the joint model, where we encourage the models to promote the mutual information between semantic representations and language representations during training. Both word-level and sentence-level features are tested in our experiments, where the sentence-level features are produced by mean-pooling over word representations. The results show that on all benchmarks, the proposed method significantly outperforms all baselines in NLG while maintaining comparable performance in NLU. For example, the proposed method has about 2x performance on BLEU, ROUGE-1, ROUGE-L, and about 3x performance on ROUGE-2 with only 1.8% degeneration on the slot F1 score (row (n)). In the joint models (e.g. rows (c)-(e)), the output from one model would be sent to another as the input, indicating that NLU and NLG have strong impact on each other. If a model produces low-quality prediction, it is intuitive that learning of two



models will easily collapse. Because the NLU baselines are strong enough, the gap for improvement is relatively small. With the connection between the models, the subtle degeneration is reasonable. Surprisingly, with the larger space to improve, the NLU model can have 3% improvement over the best-performing baseline (row (e)). In summary, by applying the proposed method, the models are able to preserve mutual information during transformation and achieve high performance in both NLU and NLG.

## 6.3 Summary

This chapter proposes an idea of referring conversations as transmission of semantics and introduces the channel capacity concept from information theory for digital communication into dialogue modeling. The proposed algorithm connects NLU and NLG models and maximizes the mutual information between semantic representations and language representations during training. The experiments conducted on three benchmark datasets consistently show that the proposed algorithm allows the NLU modules to achieve high performance and the NLG modules to obtain the significant improvement in terms of diverse metrics, demonstrating the effectiveness of the proposed method and the potential of practical usage.





## Chapter 7

# Dual Inference

Despite the effectiveness showed by the prior work, they all focused on leveraging the duality in the *training* process to obtain powerful NLU and NLG models. However, there has been little investigation on how to leverage the dual relationship into the inference stage. Considering the fast-growing scale of models in the current NLP area, such as BERT [48] and GPT-3 [49], retraining the whole models may be difficult. Due to the constraint, this paper introduces a dual inference framework, which takes the advantage of existing models from two dual tasks without re-training [33], to perform inference for each individual task regarding the duality between NLU and NLG. The contributions can be summarized as 3-fold:

- The work is the first work that proposes a dual inference framework for NLU and NLG to utilize their duality without model re-training.
- The presented framework is flexible for diverse trained models, showing the potential of practical applications and broader usage.
- The experiments on diverse benchmark datasets consistently validate the effectiveness of the proposed method.

## 7.1 Proposed Framework

With the semantics space  $\mathcal{X}$  and the natural language space  $\mathcal{Y}$ , given  $n$  data pairs  $\{(x_i, y_i)\}_{i=1}^n$  sampled from the joint space  $\mathcal{X} \times \mathcal{Y}$ , the goal of NLG is to generate corresponding utterances based on given semantics. In other words, the task is to learn a mapping function  $f(x; \theta_{x \rightarrow y})$  to transform semantic representations into natural language.

In contrast, the goal of NLU is to capture the core meaning from utterances, finding a function  $g(y; \theta_{y \rightarrow x})$  to predict semantic representations given natural language utterances. Note that in this paper, the NLU task has two parts: (1) intent prediction and (2) slot filling. Hence,  $x$  is defined as a sequence of words ( $x = \{x_i\}$ ), while semantics  $y$  can be divided into an intent  $y^I$  and a sequence of slot tags  $y^S = \{y_i^S\}$ , ( $y = (y^I, y^S)$ ). Considering that this paper focuses on the inference stage, diverse strategies can be applied to train these modules. Here we conduct a typical strategy based on maximum likelihood estimation (MLE) of the parameterized conditional distribution by the trainable parameters  $\theta_{x \rightarrow y}$  and  $\theta_{y \rightarrow x}$ .

### 7.1.1 Dual Inference

After obtaining the parameters  $\theta_{x \rightarrow y}$  and  $\theta_{y \rightarrow x}$  in the training stage, a normal inference process works as follows:

$$\begin{aligned} f(x) &= \arg \max \{ \log P(y' \mid x; \theta_{x \rightarrow y}) \}, \\ g(y) &= \arg \max \{ \log P(x' \mid y; \theta_{y \rightarrow x}) \}, \end{aligned}$$

where  $P(\cdot)$  represents the probability distribution, and  $x'$  and  $y'$  stand for model prediction. We can leverage the duality between  $f(x)$  and  $g(y)$  into the inference processes [33]. By taking NLG as an example, the core concept of dual inference is to disassemble the normal inference function into two parts: (1) inference based on the forward model  $\theta_{x \rightarrow y}$  and (2) inference based on the backward model  $\theta_{y \rightarrow x}$ . The inference process can now be rewritten

into the following:

$$f(x) \equiv \arg \max \{ \alpha \log P(y' | x; \theta_{x \rightarrow y}) + (1 - \alpha) \log P(y' | x; \theta_{y \rightarrow x}) \}, \quad (7.1)$$



where  $\alpha$  is the adjustable weight for balancing two inference components.

Based on Bayes theorem, the second term in (7.1) can be expended as follows:

$$\begin{aligned} \log P(y' | x; \theta_{y \rightarrow x}) &= \log \left( \frac{P(x | y'; \theta_{y \rightarrow x}) P(y'; \theta_y)}{P(x; \theta_x)} \right), \\ &= \log P(x | y'; \theta_{y \rightarrow x}) + \log P(y'; \theta_y) - \log P(x; \theta_x), \end{aligned}$$

where  $\theta_x$  and  $\theta_y$  are parameters for the marginal distribution of  $x$  and  $y$ . Finally, the inference process considers not only the forward pass but also the backward model of the dual task. Formally, the dual inference process of NLU and NLG can be written as:

$$\begin{aligned} f(x) &\equiv \arg \max_{y' \in \mathcal{Y}} \{ \alpha \log P(y' | x; \theta_{x \rightarrow y}) + (1 - \alpha) (\log P(x | y'; \theta_{y \rightarrow x}) \\ &\quad + \beta \log P(y'; \theta_y) - \beta \log P(x; \theta_x)) \}, \\ g(y) &\equiv \arg \max_{x' \in \mathcal{X}} \{ \alpha \log P(x' | y; \theta_{y \rightarrow x}) + (1 - \alpha) (\log P(y | x'; \theta_{x \rightarrow y}) \\ &\quad + \beta \log P(x'; \theta_x) - \beta \log P(y; \theta_y)) \}, \end{aligned}$$

where we introduce an additional weight  $\beta$  to adjust the influence of marginals. The idea behind this inference method is intuitive: the prediction from a model is reliable when the original input can be reconstructed based on it. Note that this framework is flexible for any trained models ( $\theta_{x \rightarrow y}$  and  $\theta_{y \rightarrow x}$ ), and leveraging the duality does not need any model re-training but inference.

### 7.1.2 Marginal Distribution Estimation

As derived in the previous section, marginal distributions of semantics  $P(x)$  and language  $P(y)$  are required in our dual inference method. We follow the prior work for estimating

marginals [1].

**Language Model** We train an RNN-based language model [37, 38] to estimate the distribution of natural language sentences  $P(y)$  by the cross entropy objective.

**Masked Prediction of Semantic Labels** A semantic frames  $x$  contains an intent label and some slot-value pairs; for example,  $\{Intent: "atis\_flight", fromloc.city\_name: "kansas\ city", toloc.city\_name: "los\ angeles", depart\_date.month\_name: "april\ ninth"\}$ . A semantic frame is a parallel set of discrete labels which is not suitable to model by auto-regressiveness like language modeling. Prior work [1, 2] simplified the NLU task and treated semantics as a finite number of labels, and they utilized masked autoencoders (MADE) [39] to estimate the joint distribution. However, the slot values can be arbitrary word sequences in the regular NLU setting, so MADE is no longer applicable for benchmark NLU datasets.

Considering the issue about scalability and the parallel nature, we use non-autoregressive masked models [48] to predict the semantic labels instead of MADE. The masked model is a two-layer Transformer [7] illustrated in Figure 7.1. We first encode the slot-value pairs using a bidirectional LSTM, where an intent or each slot-value pair has a corresponding encoded feature vector. Subsequently, in each iteration, we mask out some encoded features from the input and use the masked slots or intent as the targets. When estimating the density of a given semantic frame, we mask out a random input semantic feature three times and use the cumulative product of probability as the marginal distribution to predict the masked slot.

## 7.2 Experiments

To evaluate the proposed methods on a fair basis, we take two simple GRU-based models for both NLU and NLG, and the details can be found in Section 7.2.3. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively, while for NLG, the evaluation metrics include BLEU and ROUGE-(1, 2, L) scores with mul-



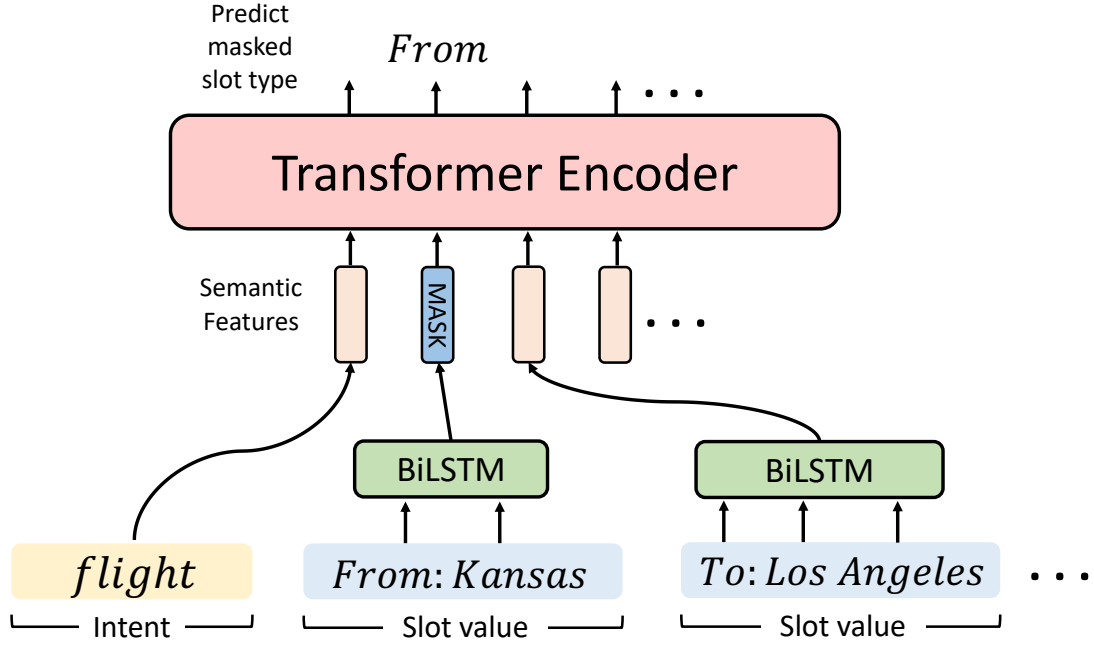


Figure 7.1: The proposed model for estimating the density of a given semantic frame.

tuple references. The hyperparameters and other training settings are reported in Section 7.2.1. The benchmark datasets conducted in our experiments are **ATIS** [19], **SNIPS** [20], and **E2E NLG** [21]. The NLG datasets are augmented into IOB-format NLU data, the augmentation methods are described in Section 2.5.2 and 2.5.3.

### 7.2.1 Training Details

In all experiments, we use mini-batch Adam as the optimizer with each batch of 48 examples on Nvidia Tesla V100. 10 training epochs were performed without early stop, the hidden size of network layers is 200, and word embedding is of size 50. The ratio of teacher forcing is set to 0.9.

### 7.2.2 Inference Details

During inference, we use beam search with beam size equal to 20. When applying dual inference, we use beam search to decode 20 possible hypotheses with the primal model (e.g. NLG). Then, we take the dual model (e.g. NLU) and the marginal models to compute the probabilities of these hypotheses in the opposite direction. Finally, we re-rank the

hypotheses using the probabilities in both directions (e.g. NLG and NLU) and select the top-1 ranked hypothesis.

To make the NLU model be able to decode different hypotheses, we need to use the auto-regressive architecture for slot filling, as described in Section 7.2.3.



### 7.2.3 Model Structure

For NLU, the model is a simple GRU [5] with a word and last output as input at each timestep  $i$  and a linear layer at the end for intent prediction based on the final hidden state:

$$o_i = \text{GRU}([w_i, o_{i-1}]).$$

The model for NLG is almost the same but with an additional encoder for encoding semantic frames, where slot-value pairs are encoded into semantic vectors for basic attention, the mean-pooled semantic vector is used as initial state. We borrow the encoder structure in [36] for our experiments. At each timestep  $i$ , the last predicted word and the aggregated semantic vector from attention are used as the input:

$$o_i = \text{GRU}([h_i^{\text{Attn}}, o_{i-1}] \mid h_{\text{mean}}).$$

### 7.2.4 Results and Analysis

Three baselines are performed for each dataset: (1) Iterative Baseline: simply training NLU and NLG iteratively, (2) Dual Supervised Learning (Chapter 4) [1], and (3) Joint Baseline: the output from one model is sent to another as in (Chapter 5) [2]<sup>1</sup>. In joint baselines, the outputs of NLU are intent and IOB-slot tags, whose modalities are different from the NLG input, so a simple matching method is performed (see Section 2.5.2 and 2.5.3).

For each trained baseline, the proposed dual inference technique is applied. The inference details are reported in Section 8.3.3. We try two different approaches of searching

<sup>1</sup>In our NLU setting, it is infeasible to flow the gradients through the loop for training the models jointly.

Learning Scheme	NLU		NLG			
	Accuracy	F1	BLEU	R-1	R-2	R-L
<b>ATIS</b>						
Iterative Baseline	84.10	94.26	16.08	35.10	11.94	33.73
+ DualInf( $\alpha=0.5, \beta=0.5$ )	85.07	93.84	<b>17.38</b>	<b>36.40</b>	<b>13.33</b>	<b>35.09</b>
+ DualInf( $\alpha^*, \beta^*$ )	<b>85.57</b>	<b>94.63</b>	16.06	35.19	11.93	33.75
Dual Supervised Learning	82.98	94.85	16.98	38.83	15.56	37.50
+ DualInf( $\alpha=0.5, \beta=0.5$ )	83.68	94.89	<b>20.69</b>	<b>40.62</b>	<b>17.72</b>	<b>39.31</b>
+ DualInf( $\alpha^*, \beta^*$ )	<b>84.26</b>	<b>95.32</b>	17.05	38.82	15.57	37.42
Joint Baseline	81.44	90.37	21.00	39.70	18.91	38.48
+ DualInf( $\alpha=0.5, \beta=0.5$ )	81.21	88.42	<b>22.60</b>	<b>41.19</b>	<b>20.24</b>	<b>39.88</b>
+ DualInf( $\alpha^*, \beta^*$ )	<b>85.88</b>	<b>90.66</b>	20.67	39.41	18.68	38.16

Table 7.1: The model performance for the ATIS dataset. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). All reported numbers are averaged over three different runs.

Learning Scheme	NLU		NLG			
	Accuracy	F1	BLEU	R-1	R-2	R-L
<b>SNIPS</b>						
Iterative Baseline	96.58	96.67	15.49	34.32	13.75	33.26
+ DualInf( $\alpha=0.5, \beta=0.5$ )	<b>97.07</b>	<b>96.70</b>	<b>16.90</b>	<b>35.43</b>	<b>15.18</b>	<b>34.41</b>
+ DualInf( $\alpha^*, \beta^*$ )	96.88	96.76	15.46	34.21	13.78	33.14
Dual Supervised Learning	96.83	96.71	15.96	36.69	15.39	35.73
+ DualInf( $\alpha=0.5, \beta=0.5$ )	<b>96.88</b>	<b>96.80</b>	<b>18.07</b>	<b>37.63</b>	<b>16.75</b>	<b>36.67</b>
+ DualInf( $\alpha^*, \beta^*$ )	95.34	96.68	16.08	36.97	15.62	36.04
Joint Baseline	97.18	94.57	17.15	36.32	15.68	35.36
+ DualInf( $\alpha=0.5, \beta=0.5$ )	<b>97.27</b>	95.59	<b>18.56</b>	37.87	17.25	36.90
+ DualInf( $\alpha^*, \beta^*$ )	95.54	<b>96.06</b>	18.26	<b>38.16</b>	<b>17.70</b>	<b>37.40</b>

Table 7.2: The model performance for the SNIPS dataset. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). All reported numbers are averaged over three different runs.

inference parameters ( $\alpha$  and  $\beta$ ):

- DualInf( $\alpha=0.5, \beta=0.5$ ): simply uses  $\alpha=0.5$  and  $\beta=0.5$  to balance the effect of backward inference and the influence of the marginal distributions.
- DualInf( $\alpha^*, \beta^*$ ): uses the best parameters  $\alpha=\alpha^*$  and  $\beta=\beta^*$  searched by using validation set for intent prediction, slot filling, language generation individually. The parameters  $\alpha$  and  $\beta$  ranged from 0.0 to 1.0, with a step of 0.1; hence for each task,

Learning Scheme	NLU		NLG			
	Accuracy	F1	BLEU	R-1	R-2	R-L
<b>E2E NLG</b>						
Iterative Baseline	-	94.25	24.98	44.60	19.40	37.99
+ DualInf( $\alpha=0.5, \beta=0.5$ )	-	94.29	25.34	44.82	19.73	38.23
+ DualInf( $\alpha^*, \beta^*$ )	-	<b>94.55</b>	<b>25.35</b>	<b>44.87</b>	<b>19.74</b>	<b>38.30</b>
Dual Supervised Learning	-	94.49	24.73	45.74	19.60	39.91
+ DualInf( $\alpha=0.5, \beta=0.5$ )	-	<b>94.53</b>	<b>25.40</b>	<b>46.25</b>	<b>20.18</b>	<b>40.42</b>
+ DualInf( $\alpha^*, \beta^*$ )	-	94.47	24.67	45.71	19.56	39.88
Joint Baseline	-	93.51	25.19	44.80	19.59	38.20
+ DualInf( $\alpha=0.5, \beta=0.5$ )	-	93.43	<b>25.57</b>	45.11	<b>19.90</b>	38.56
+ DualInf( $\alpha^*, \beta^*$ )	-	<b>93.88</b>	25.54	<b>45.17</b>	19.89	<b>38.61</b>

Table 7.3: The model performance for the E2E NLG dataset. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). All reported numbers are averaged over three different runs.

there are 121 pairs of  $(\alpha, \beta)$ .

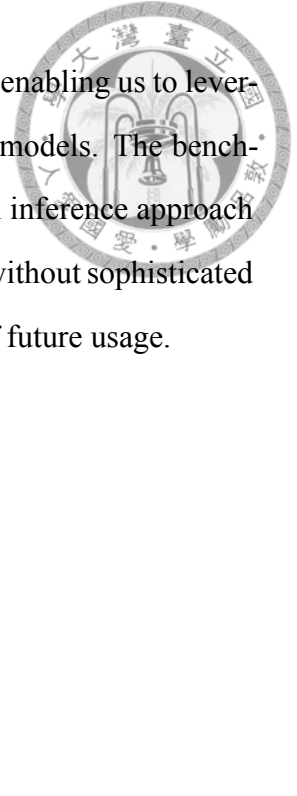
The results are shown in Table 7.1, 7.2, and 7.3. For ATIS, all NLU models achieve the best performance by selecting the parameters for intent prediction and slot filling individually. For NLG, the models with  $(\alpha=0.5, \beta=0.5)$  outperform the baselines and the ones with  $(\alpha^*, \beta^*)$ , probably because of the discrepancy between the validation set and the test set. In the results of SNIPS, for the models mainly trained by standard supervised learning (iterative baseline and dual supervised learning), the proposed method with  $(\alpha=0.5, \beta=0.5)$  outperform the others in both NLU and NLG. However, the model trained with the connection between NLU and NLG behaves different, which performs best on slot F-1 and ROUGE with  $(\alpha^*, \beta^*)$  and performs best on intent accuracy and ROUGE with  $(\alpha=0.5, \beta=0.5)$ . For E2E NLG, the results show a similar trend as ATIS, better NLU results with  $(\alpha^*, \beta^*)$  in NLU and better NLG performance with  $(\alpha=0.5, \beta=0.5)$ .

In summary, the proposed dual inference technique can consistently improve the performance of NLU and NLG models trained by different learning algorithms, showing its generalization to multiple datasets/domains and flexibility of diverse training baselines. Furthermore, for the models learned by standard supervised learning, simply picking the inference parameters  $(\alpha=0.5, \beta=0.5)$  would possibly provide improvement on performance.



## 7.3 Summary

This work introduces a dual inference framework for NLU and NLG, enabling us to leverage the duality between the tasks without re-training the large-scale models. The benchmark experiments demonstrate the effectiveness of the proposed dual inference approach for both NLU and NLG trained by different learning algorithms even without sophisticated parameter search on different datasets, showing the great potential of future usage.







## Chapter 8

# Dual Finetuning

Recent progress in training large-scale language models (e.g., GPT, GPT-2) [50, 51] has inspired a new wave of approaches for many natural language processing tasks. Finetuning pre-trained language models and transferring the learned knowledge to target tasks have become basics in NLP research. In this chapter, we study how to finetune the pretrained language models for the two dual tasks.

### 8.1 Pretrained Language Models

Recently in NLP research, pretrained models have become popular for various tasks. ELMo [52] is a large-scale pre-trained bi-LSTM sentence representation learning framework, open-sourced by AllenNLP [53]. Masked language modeling and next sentence prediction are the well-known pretraining objectives, these models including BERT [54] and RoBERTa [55], are suitable of finetuning for classification problems. These models are often based on bidirectional Transformer encoders. On the other hand, GPT [50], GPT-2 [51], and XLNet [56] take auto-regressive language modeling as one of the pretraining objectives. The auto-regressive language modeling objective makes the models suitable of generating sequences of tokens. Besides singleton models, sequence-to-sequence models including BART [57] and T5 [58] combine the advantages from both bidirectional and auto-regressive Transformers, which makes them applicable to a very wide range of end tasks.

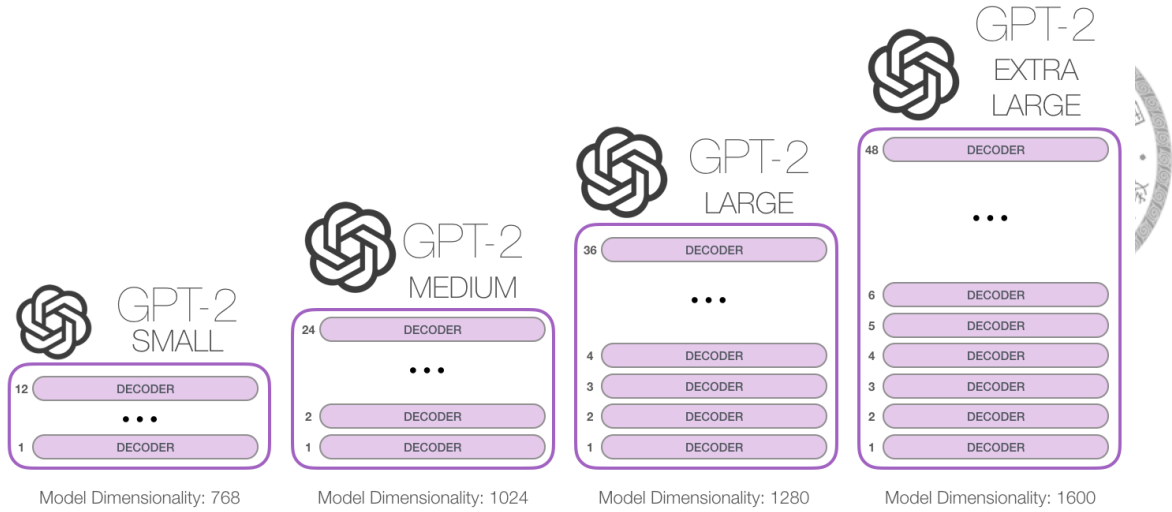


Figure 8.1: Source: <https://jalamar.github.io/illustrated-gpt2/>

## 8.2 Generative Pretrained Transformer 2 (GPT-2)

In the proposed method, we propose to use GPT-2 as the backbone [51] regarding its auto-regressiveness. GPT-2 is the successor to GPT [50], which also uses the layered Transformer decoder structure [7], which is illustrated in Chapter 2. In contrast to GPT, GPT-2 uses 50,257 Byte-Pair Encoding (BPE) tokens and places the Layer Norm before the Masked Multi-Head component, an additional Layer Norm is added after the final block. The maximum sequence length is increased from 512 to 1024; the mini-batch size during pretraining is increased from 64 to 512. Four pretrained GPT-2 models with different numbers of decoder blocks are available (small, medium, large, extra-large), as illustrated in Figure 8.1. The largest one has 48 blocks with model dimension of 1600, resulting in a total number of 1.5 billion model parameters. The pretraining datasets for GPT-2 is also different to that for GPT. The authors further created a new web scrape which emphasizes document quality, they only scraped web pages which have been curated/filtered by humans. The resulting dataset, WebText, has over 8 million documents for a total of 40 GB of text.

### 8.3 Objective Design

In this work, we aim to solve the two dual tasks in a single model, in this case, the input space and the output space need to completely cover both the language space and the semantic space. Though BERT and other bidirectional models are good at classification tasks (NLU), they cannot generate sequences auto-regressively (NLG). Therefore we instead model both tasks as text generation, where we parse intent labels (sequence classification) and slot labels (token classification) into a semantic sequence as target. At first, we directly concatenate the intent label and the IOB-scheme tags, for example:

[ atis\_flight O O B-fromloc.city\_name O B-toloc.city\_name I-toloc.city\_name ],

where "atis\_flight" is the intent label and the other tokens are slot labels. This method is straight-forward and there is no need for further post-processing, take NLU for example, we definitely know the number of input word tokens hence we can take the same number of decoded slot tags. However, the input slot tags does not have slot-value information, which means it is impossible to reconstruct the original utterance. Another issue comes from tokenization, since we choose pretrained GPT-2 models as the backbone models, in order to leverage the learned knowledge inside the models, we need to follow the tokenization methods and input space. GPT-2 models are pretrained on free-text dataset, the slot tags like "B-fromloc.city\_name" would be parsed into many subword tokens, which would accordingly make the input and target sequences lengthy and hard to optimize. Therefore we extract the slot-value pairs and drop the I- and B- prefixes to form a semantic frame:

[ atis\_flight; fromloc.city\_name: charlotte; toloc.city\_name: las vegas; ],

where ";," and ":" are used to separate the labels.

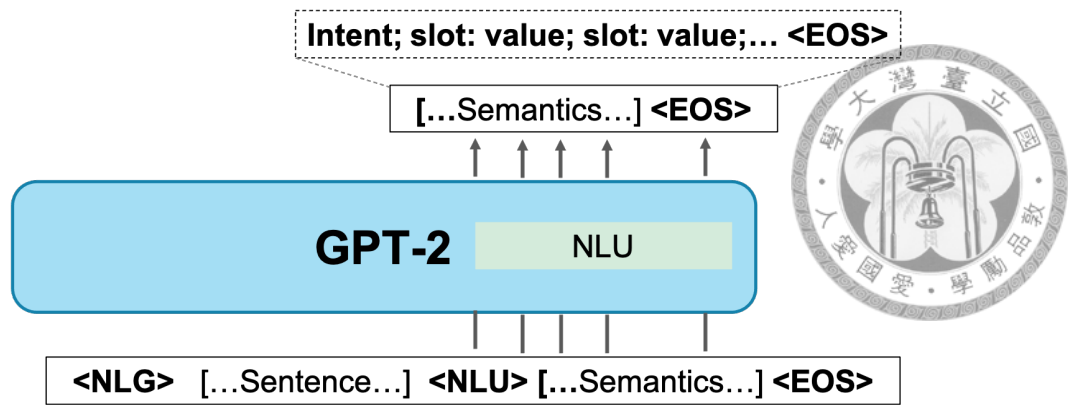


Figure 8.2: The objective design of De-Training, which the model is finetuned for a task at one time. <NLU>, <NLG>, and <EOS> are the special tokens.

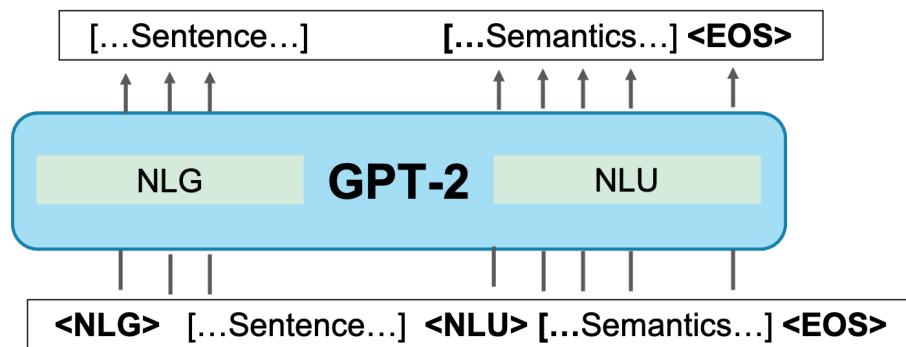


Figure 8.3: The objective design of Co-Training, which the model is finetuned for both tasks at one time. <NLU>, <NLG>, and <EOS> are the special tokens.

### 8.3.1 De-Training

We first propose De-Training objective, which we construct the input sequence by concatenate the utterance and the semantic sequence with special tokens <NLG> before the utterance and <NLU> before the semantic sequence, respectively. The input sequence is also attached with a special token <EOS> at the end to indicate the end-of-sentence, as illustrated in Figure 8.2. The training scheme is standard causal language modeling, in this way, the model would learn to perform NLU when seeing <NLU> based on the given sentence, and vice versa.

### 8.3.2 Co-Training

We also propose another objective, called Co-Training. The idea of Co-Training is to train the two dual tasks at one time, hence we could directly learn language modeling of whole

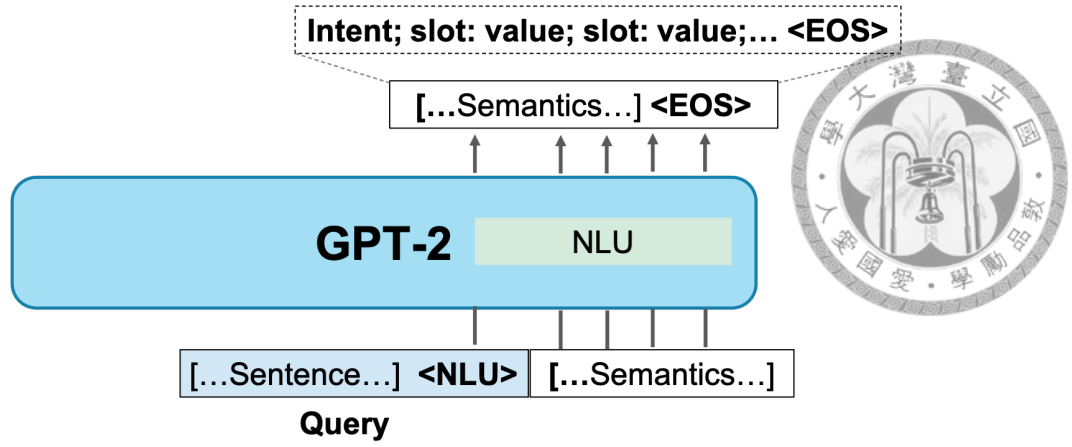


Figure 8.4: The illustration of inference of NLU, the input sentence is attached with a **<NLU>** token to enforce the model to generate semantic tokens according to the given context.

input sequences (Figure 8.3) to realize the idea.

### 8.3.3 Inference

After finetuning, the models are expected to recognize the input context and the special tokens and accordingly perform the target tasks. For example, we could attach the special token **<NLU>** at the end of the sentence to construct the input query. The input query is then fed into the model and the model accordingly generate the sequence auto-regressively, as illustrated in Figure 8.4 .

## 8.4 Experiments

The benchmark datasets conducted in our experiments are **ATIS** [19], and **SNIPS** [20]. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%). The implementation is based on *HuggingFace Transformers*<sup>1</sup> library and PyTorch. 100 training epochs were performed on NVidia Tesla V100 GPUs.

The experimental results are shown in Table 8.1, from the table we can see that for ATIS dataset, the proposed dual finetuning (Row (d)) outperforms all the previously-proposed

<sup>1</sup><https://huggingface.co/transformers/>

Learning Scheme		NLU		NLG			
		Accuracy	F1	BLEU	R-1	R-2	R-L
<b>ATIS</b>							
(a)	Dual Supervised Learning	83.02	94.73	16.72	37.89	14.60	36.53
(b)	Joint Model	80.61	91.26	17.26	38.10	14.69	36.73
(c)	(b) + MI(semantics, word)	88.15	93.75	24.46	42.92	23.01	41.78
(d)	<b>Dual Finetuning</b>	<b>96.88</b>	<b>97.54</b>	<b>41.35</b>	<b>69.65</b>	<b>51.87</b>	<b>68.56</b>
<b>SNIPS</b>							
(e)	Dual Supervised Learning	<b>97.39</b>	<b>96.35</b>	15.90	39.85	16.39	38.69
(f)	Joint Model	97.32	94.56	17.19	38.59	16.36	37.53
(g)	(h) + MI(semantics, word)	97.02	94.25	19.30	42.20	19.66	40.83
(h)	<b>Dual Finetuning</b>	<b>97.37</b>	90.08	<b>31.37</b>	<b>67.76</b>	<b>47.42</b>	<b>66.72</b>

Table 8.1: The backbones of dual finetuning (Row (d) and (h)) are GPT-2 (Large) and de-training. For NLU, accuracy and F1 measure are reported for intent prediction and slot filling respectively. The NLG performance is reported by BLEU, ROUGE-1, ROUGE-2, and ROUGE-L of models (%).

methods on all the metrics of NLU and NLG. Especially for NLG, dual finetuning could achieve 69% and 63% improvement on BLEU score and ROUGE-1 score respectively, and over 100% improvement on ROUGE-2 score. The superior performance of NLG is reasonable, because the GPT-2 models are good at text generation. For SNIPS dataset, the proposed methods can achieve comparable performance of NLU and likewise over 60% improvement on BLEU score and ROUGE-1 score. The experimental results not only prove that the proposed methods are effective, but also show the potential of solving two dual tasks in a single model.

When we were designing the training objectives, we presume that Co-Training would outperform De-Training since it learn two tasks at one time, which may have better data efficiency. However, as shown in Table 8.2 and Figure 8.5, Co-Training only works better with smaller models (small and medium sizes). However, Large-size models with De-Training are the best-performing ones and comparable to previous methods. We presume the reason is that learning toward two dual tasks at one time would confuse the models and make the optimization out of focus. When utilizing small models, better data efficiency does help, but larger models come with more difficulty of optimization.



Learning Scheme		NLU		NLG			
		Accuracy	F1	BLEU	R-1	R-2	R-L
<b>ATIS (Dual Finetuning)</b>							
(a)	GPT-2 Small	19.44	28.59	8.32	14.85	10.51	14.70
(b)	+ Co-Training	42.13	71.54	8.83	15.93	9.44	15.39
(c)	GPT-2 Medium	25.23	36.54	20.14	33.45	24.01	33.01
(d)	+ Co-Training	35.30	62.18	23.64	39.35	25.38	37.90
(e)	GPT-2 Large	<b>96.88</b>	<b>97.54</b>	<b>41.35</b>	<b>69.65</b>	<b>51.87</b>	<b>68.56</b>
(f)	+ Co-Training	68.63	81.21	39.15	63.14	45.26	62.19
<b>SNIPS (Dual Finetuning)</b>							
(g)	GPT-2 Small	17.56	28.09	10.64	21.89	16.69	21.64
(h)	+ Co-Training	17.11	29.53	13.38	31.30	18.82	30.40
(i)	GPT-2 Medium	33.68	46.96	28.79	64.90	48.21	64.35
(j)	+ Co-Training	46.03	70.96	13.42	30.83	19.83	29.88
(k)	GPT-2 Large	<b>97.37</b>	<b>90.08</b>	<b>31.37</b>	<b>67.76</b>	<b>47.42</b>	<b>66.72</b>
(l)	+ Co-Training	49.70	66.02	21.28	44.83	29.13	43.98

Table 8.2: The comparison of overall performance of finetuning GPT-2 of different sizes. The default training scheme is de-training, so the rows like Row (a) is the performance of de-training while Row (b) is the performance of co-training.

## 8.5 Summary

In this work, we study how to finetune the pretrained language models for the two dual tasks. We propose to employ GPT-2 models as the backbone and model both NLU and NLG as text generation. The experiments on the benchmark datasets demonstrate that the proposed methods are effective especially for NLG, motivating the potential research directions in this area.

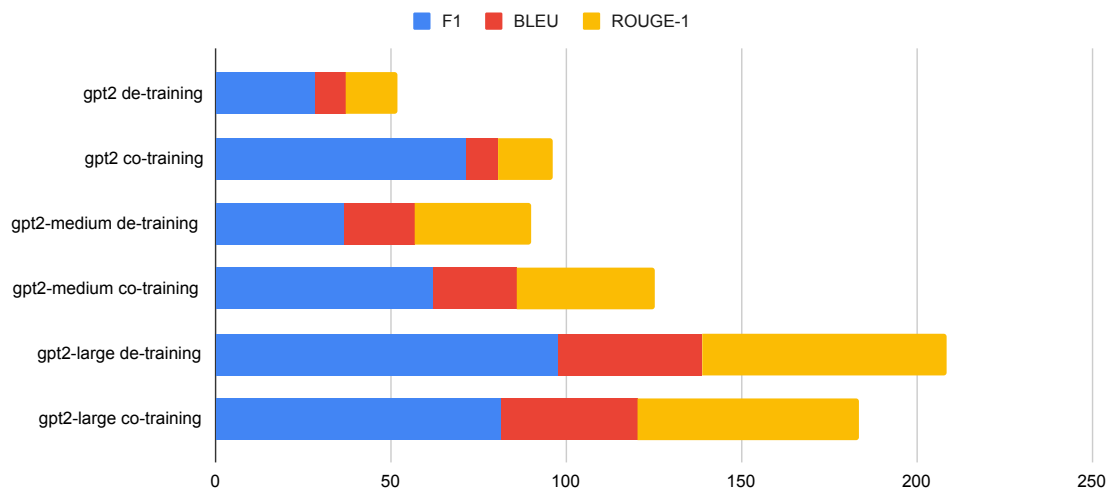


Figure 8.5: The comparison of overall performance of finetuning GPT-2 of different sizes, *de-training* is to train a task at one time and *co-training* is to train both tasks at one time. The results show that Co-Training only works better with smaller models.



## Chapter 9

# Conclusion and Future Work

### 9.1 Challenges

One of the main challenges of this research direction lies in data, in our experiments, we augment certain NLU datasets into NLG datasets, and vice versa. However, not every NLU/NLG dataset is suitable to be augmented into its dual task. Broadly speaking, the general idea of natural language understanding is to capture the core concept of given utterances, as our definition in Chapter 2. NLU is a huge family of tasks, which makes it have different configurations of semantic labels from task to task. For example, natural language inference (NLI) is the task of determining whether a "hypothesis" is true (entailment), false (contradiction), or undetermined (neutral) given a "premise". NLI is a single-label classification NLU task, which has three different semantic label, it is intuitive that such data could not be used as NLG data. Another example is Part-of-Speech (POS) tagging, which is the task of marking up a word in a text corpus as corresponding to a particular part of speech, based on both its definition and its context. POS tagging is a sequence tagging task, for example, a verb token in the base form "play" will be annotated as "VB". Since there are countless verb-base tokens, it is impossible to predict "play" given a "VB". Basically, the semantic annotation should have high fidelity and rich semantic information, which enables reverse restoration (NLG).

As described above, NLU is a huge family of tasks, given an utterance, how human annotate the semantic labels makes huge difference. Such task nature makes the dual rela-

tionship between NLU and NLG different from other dual task pair like ASR v.s. TTS or NMT v.s. NMT. Moreover, since NLU always requires human annotations, technically it is infeasible to perform "fully" unsupervised learning. In our work (Chapter 5), we failed learning two models by the training cycles with only one part of the data, like only text or only semantics. Even if we succeeded, we were still using human-annotated semantic labels, making it *semi-supervised learning*. Unlike other dual task pairs, the dual relationship is asymmetrical not only because NLU is a *strong* many-to-one problem and NLG is an one-to-many problem, but also because of the data modality. In NMT v.s. NMT, the data  $x$  and  $y$  are both text tokens, while in terms of NLU v.s. NLG, semantics do not have sequential nature, making it hard to design neural models in dual form.

## 9.2 Conclusion

This main goal of this dissertation is to investigate the structural duality between NLU and NLG. In this thesis, we present four consecutive studies, each focuses on different aspects of learning and data settings. First, we exploits the duality between NLU and NLG and introduces it into the learning objective as the regularization term. Moreover, expert knowledge is incorporated to design suitable approaches for estimating data distribution. Second, we further propose a joint learning framework, which provides flexibility of incorporating not only supervised but also unsupervised learning algorithms and enables the gradient to propagate through two modules seamlessly. Third, we study how to enhance the joint framework by mutual information maximization. Fourth, since above works exploit the duality in the training stage, hence we make a step forward to leverage the duality in the inference stage. Lastly, we finetune the pretrained language models on the two dual tasks and achieve the goal of solving two dual tasks in a single model. Each work presents a new model and learning framework exploiting the duality in different manners. Together, this dissertation explores a new research direction of exploiting the duality between language understanding and generation.

## 9.3 Future Work

### 9.3.1 Advanced Learning Algorithms

Though the proposed framework (Chapter 5) provides possibility of training two models in a fully unsupervised manner, it was found unstable and hard to optimize from our experiments. Therefore, better dual learning algorithms and leveraging pretrained models and other learning techniques, such as adversarial learning, are worthy to explore for improving the proposed framework. [59] proposed a model coupling NLU and NLG with a latent variable representing the shared intent between natural language and formal representations.

### 9.3.2 Connection to Pragmatics

Recently, [35] improved models for conditional text generation using techniques from computational pragmatics. The techniques formulated language production as a game between speakers and listeners, where a speaker should generate text which a listener can use to correctly identify the original input the text describes. The proposed idea in [35] is actually analogous to the proposed dual learning algorithms [1, 2], however we develop the idea from the perspective of learning while they tackle the problem by pragmatic knowledge. A branch of computational pragmatics is to study the interaction between speakers and listeners.









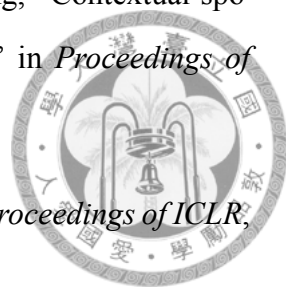
# Bibliography

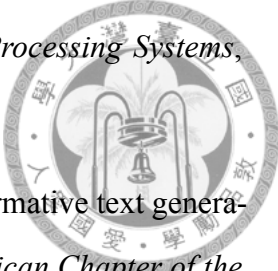
- [1] S.-Y. Su, C.-W. Huang, and Y.-N. Chen, “Dual supervised learning for natural language understanding and generation,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5472–5477, 2019.
- [2] S.-Y. Su, C.-W. Huang, and Y.-N. Chen, “Towards unsupervised language understanding and generation by joint dual learning,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.
- [3] S.-Y. Su, Y.-S. Chuang, and Y.-N. Chen, “Dual inference for improving language understanding and generation,” in *Findings of the Association for Computational Linguistics: EMNLP 2020*, (Online), pp. 4930–4936, Association for Computational Linguistics, Nov. 2020.
- [4] E. J. L., “Finding structure in time,” *Cognitive Science*, vol. 14, no. 2, pp. 179–211.
- [5] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.
- [6] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, “Convolutional sequence to sequence learning,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1243–1252, JMLR. org, 2017.
- [7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, pp. 5998–6008, 2017.


- 
- [8] T.-H. Wen, M. Gasic, N. Mrksic, L. M. Rojas-Barahona, P.-H. Su, S. Ultes, D. Vandyke, and S. Young, “A network-based end-to-end trainable task-oriented dialogue system,” in *Proceedings of EACL*, pp. 438–449, 2017.
- [9] A. Bordes, Y.-L. Boureau, and J. Weston, “Learning end-to-end goal-oriented dialog,” in *Proceedings of ICLR*, 2017.
- [10] B. Dhingra, L. Li, X. Li, J. Gao, Y.-N. Chen, F. Ahmed, and L. Deng, “Towards end-to-end reinforcement learning of dialogue agents for information access,” in *Proceedings of ACL*, pp. 484–495, 2017.
- [11] X. Li, Y.-N. Chen, L. Li, J. Gao, and A. Celikyilmaz, “End-to-end task-completion neural dialogue systems,” in *Proceedings of The 8th International Joint Conference on Natural Language Processing*, 2017.
- [12] G. Tur and R. De Mori, *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons, 2011.
- [13] D. Hakkani-Tür, G. Tür, A. Celikyilmaz, Y.-N. Chen, J. Gao, L. Deng, and Y.-Y. Wang, “Multi-domain joint semantic frame parsing using bi-directional rnn-lstm,” in *Proceedings of INTERSPEECH*, pp. 715–719, 2016.
- [14] B. Peng, X. Li, J. Gao, J. Liu, K.-F. Wong, and S.-Y. Su, “Deep dyna-q: Integrating planning for task-completion dialogue policy learning,” *arXiv preprint arXiv:1801.06176*, 2018.
- [15] S.-Y. Su, X. Li, J. Gao, J. Liu, and Y.-N. Chen, “Discriminative deep dyna-q: Robust planning for dialogue policy learning,” in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3813–3823, 2018.
- [16] T.-H. Wen, M. Gasic, N. Mrkšić, P.-H. Su, D. Vandyke, and S. Young, “Semantically conditioned lstm-based natural language generation for spoken dialogue systems,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1711–1721, 2015.

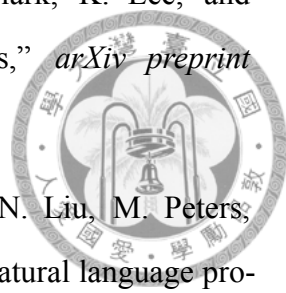


- 
- [17] S.-Y. Su, K.-L. Lo, Y.-T. Yeh, and Y.-N. Chen, “Natural language generation by hierarchical decoding with linguistic patterns,” in *Proceedings of The 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2018.
- [18] S.-Y. Su and Y.-N. Chen, “Investigating linguistic pattern ordering in hierarchical natural language generation,” in *Proceedings of 7th IEEE Workshop on Spoken Language Technology*, pp. 779–786, 2018.
- [19] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, “The atis spoken language systems pilot corpus,” in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*, 1990.
- [20] A. Coucke, A. Saade, A. Ball, T. Bluche, A. Caulier, D. Leroy, C. Doumouro, T. Gisselbrecht, F. Caltagirone, T. Lavril, *et al.*, “Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces,” *arXiv preprint arXiv:1805.10190*, 2018.
- [21] J. Novikova, O. Dušek, and V. Rieser, “The e2e dataset: New challenges for end-to-end generation,” in *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pp. 201–206, 2017.
- [22] Y.-N. Chen, D. Hakkani-Tur, G. Tur, A. Celikyilmaz, J. Gao, and L. Deng, “Knowledge as a teacher: Knowledge-guided structural attention networks,” *arXiv preprint arXiv:1609.03286*, 2016.
- [23] Y.-N. Chen, M. Sun, A. I. Rudnicky, and A. Gershman, “Leveraging behavioral patterns of mobile applications for personalized spoken language understanding,” in *Proceedings of ICMI*, pp. 83–86, 2015.
- [24] M. Sun, Y.-N. Chen, and A. I. Rudnicky, “An intelligent assistant for high-level task understanding,” in *Proceedings of IUI*, pp. 169–174, 2016.

- 
- [25] Y. Shi, K. Yao, H. Chen, Y.-C. Pan, M.-Y. Hwang, and B. Peng, “Contextual spoken language understanding using recurrent neural networks,” in *Proceedings of ICASSP*, pp. 5271–5275, 2015.
- [26] J. Weston, S. Chopra, and A. Bordes, “Memory networks,” in *Proceedings of ICLR*, 2015.
- [27] P.-C. Chen, T.-C. Chi, S.-Y. Su, and Y.-N. Chen, “Dynamic time-aware attention to speaker roles and contexts for spoken language understanding,” in *Proceedings of ASRU*, 2017.
- [28] S.-Y. Su, P.-C. Yuan, and Y.-N. Chen, “How time matters: Learning time-decay attention for contextual spoken language understanding in dialogues,” in *Proceedings of NAACL-HLT*, 2018.
- [29] T.-H. Wen, M. Gasic, D. Kim, N. Mrksic, P.-H. Su, D. Vandyke, and S. Young, “Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking,” in *Proceedings of SIGDIAL*, pp. 275–284, 2015.
- [30] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Proceedings of NIPS*, pp. 3104–3112, 2014.
- [31] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, “Google’s neural machine translation system: Bridging the gap between human and machine translation,” *arXiv preprint arXiv:1609.08144*, 2016.
- [32] A. Tjandra, S. Sakti, and S. Nakamura, “Listening while speaking: Speech chain by deep learning,” in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 301–308, IEEE, 2017.
- [33] Y. Xia, T. Qin, W. Chen, J. Bian, N. Yu, and T.-Y. Liu, “Dual supervised learning,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3789–3798, JMLR. org, 2017.

- 
- [34] D. He, Y. Xia, T. Qin, L. Wang, N. Yu, T.-Y. Liu, and W.-Y. Ma, “Dual learning for machine translation,” in *Advances in Neural Information Processing Systems*, pp. 820–828, 2016.
- [35] S. Shen, D. Fried, J. Andreas, and D. Klein, “Pragmatically informative text generation,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4060–4067, ACL, June 2019.
- [36] S. Zhu, R. Cao, and K. Yu, “Dual learning for semi-supervised natural language understanding,” *arXiv preprint arXiv:2004.12299*, 2020.
- [37] T. Mikolov, M. Karafiát, L. Burget, J. Černocký, and S. Khudanpur, “Recurrent neural network based language model,” in *Eleventh annual conference of the international speech communication association*, 2010.
- [38] M. Sundermeyer, R. Schlüter, and H. Ney, “Lstm neural networks for language modeling,” in *Thirteenth annual conference of the international speech communication association*, 2012.
- [39] M. Germain, K. Gregor, I. Murray, and H. Larochelle, “Made: Masked autoencoder for distribution estimation,” in *International Conference on Machine Learning*, pp. 881–889, 2015.
- [40] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, “Policy gradient methods for reinforcement learning with function approximation,” in *Advances in neural information processing systems*, pp. 1057–1063, 2000.
- [41] Y. Bengio, N. Léonard, and A. Courville, “Estimating or propagating gradients through stochastic neurons for conditional computation,” *arXiv preprint arXiv:1308.3432*, 2013.
- [42] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

- 
- [43] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan, “A diversity-promoting objective function for neural conversation models,” *CoRR*, vol. abs/1510.03055, 2015.
- [44] I. Belghazi, S. Rajeswar, A. Baratin, R. D. Hjelm, and A. C. Courville, “MINE: mutual information neural estimation,” *CoRR*, vol. abs/1801.04062, 2018.
- [45] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, “Learning deep representations by mutual information estimation and maximization,” *arXiv preprint arXiv:1808.06670*, 2018.
- [46] Y.-T. Yeh and Y.-N. Chen, “Qainfomax: Learning robust question answering system by mutual information maximization,” *arXiv preprint arXiv:1909.00215*, 2019.
- [47] C.-W. Goo, G. Gao, Y.-K. Hsu, C.-L. Huo, T.-C. Chen, K.-W. Hsu, and Y.-N. Chen, “Slot-gated modeling for joint slot filling and intent prediction,” in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 753–757, 2018.
- [48] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [49] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, “Language models are few-shot learners,” *arXiv preprint arXiv:2005.14165*, 2020.
- [50] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving language understanding by generative pre-training,” 2018.
- [51] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

- 
- [52] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” *arXiv preprint arXiv:1802.05365*, 2018.
- [53] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. Liu, M. Peters, M. Schmitz, and L. Zettlemoyer, “Allennlp: A deep semantic natural language processing platform,” *arXiv preprint arXiv:1803.07640*, 2018.
- [54] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.
- [55] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” *arXiv preprint arXiv:1907.11692*, 2019.
- [56] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, “Xlnet: Generalized autoregressive pretraining for language understanding,” *Advances in neural information processing systems*, vol. 32, 2019.
- [57] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, “Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension,” *arXiv preprint arXiv:1910.13461*, 2019.
- [58] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *arXiv preprint arXiv:1910.10683*, 2019.
- [59] B.-H. Tseng, J. Cheng, Y. Fang, and D. Vandyke, “A generative model for joint natural language understanding and generation,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, (Online), pp. 1795–1807, Association for Computational Linguistics, July 2020.