UNIVERSITÀ
DI TRENTO

DEPARTMENT OF INFORMATION ENGINEERING AND COMPUTER SCIENCE

ICT International Doctoral School

Low-Resource Natural Language Understanding in Task-Oriented Dialogue

Samuel Louvan

Advisor:

Bernardo Magnini Fondazione Bruno Kessler

Thesis Committee:

Roberto Basili University of Roma, Tor Vergata

Raffaella Bernardi University of Trento

Barbara Plank IT University of Copenhagen

2022

Abstract

Task-oriented dialogue (ToD) systems need to interpret the user's input to understand the user's needs (*intent*) and corresponding relevant information (*slots*). This process is performed by a Natural Language Understanding (NLU) component, which maps the text utterance into a semantic frame representation, involving two subtasks: intent classification (*text classification*) and slot filling (*sequence tagging*). Typically, new domains and languages are regularly added to the system to support more functionalities. Collecting domain-specific data and performing fine-grained annotation of large amounts of data every time a new domain and language is introduced can be expensive. Thus, developing an NLU model that generalizes well across domains and languages with less labeled data (low-resource) is crucial and remains challenging.

This thesis focuses on investigating transfer learning and data augmentation methods for low-resource NLU in ToD. Our first contribution is a study of the potential of non-conversational text as a source for transfer. Most transfer learning approaches assume labeled conversational data as the source task and adapt the NLU model to the target task. We show that leveraging similar tasks from non-conversational text improves performance on target slot filling tasks through multi-task learning in low-resource settings. Second, we propose a set of lightweight augmentation methods that apply data transformation on token and sentence levels through slot value substitution and syntactic manipulation. Despite its simplicity, the performance is comparable to deep learning-based augmentation models, and it is effective on six languages on NLU tasks. Third, we investigate the effectiveness of domain adaptive pre-training for zero-shot cross-lingual NLU. In terms of overall performance, continued pre-training in English is effective across languages. This result indicates that the domain knowledge learned in English is transferable to other languages. In addition to that, domain similarity is essential. We show that intermediate

pre-training data that is more similar – in terms of data distribution – to the target dataset yields better performance.

Keywords Task-Oriented Dialogue, Natural Language Understanding, Limited Labeled Data, Slot Filling, Intent Classification

Acknowledgments

The work in this Ph.D. thesis is possible because of the help of people I have met during my Ph.D. study. I want to thank my advisor, Bernardo Magnini, for taking the chance on me and being willing to guide me throughout my Ph.D. Bernardo has been very patient and supportive of me. He has given me the freedom to explore my ideas and consistently gives me critical feedback whenever needed. I want to thank all the committee and thesis reviewers, especially Barbara Plank and Yannis Konstas, who provided valuable feedback and comments to improve the writing of this thesis.

I am indebted to Clara Vania, who is willing to discuss any technical topics related to my research. She read almost all of my papers before I submitted them. Thank you for your kindness and for always being there through my ups and downs. Thank you also for feeding me every day with your tasty lunch and dinner. I hope you can learn to cook Italian dishes in the future, especially the dessert that I miss.

The internship experience in Amazon Alexa was a positive "distraction" in my Ph.D. For this, I want to thank Angeliki Metallinou and Maryam Fazel-Zarandi. The internship was a challenging experience during the pandemic, so I want to thank my mentor, Alessandra Cervone. Alessandra has been kind and helpful through my training and guided me in my research internship. The internship experience has taught me a lot about working effectively in teams and given me different experiences doing applied research in industry settings.

I want to thank Silvia Casola for being a fantastic co-author, colleague, and friend with whom I can share a joke. Thank you to all colleagues and friends in FBK: Yi-Ling, Vevake, Serra, Alina, Amir, Surafel, Mattia, Flor, Marco Gaido, and Sara. Finally, I want to thank my family. My mother, Essy, and my sister, Darlene, constantly support me through their prayers and unconditional love. Looking back, it's hard for me to believe

that I can endure all the difficulties, especially during the pandemic, being far away from my family. For that, thank you to all of you.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Building a dialogue system that can engage a real-world conversation with a human has been one of the ultimate and long-standing goals of Artificial Intelligence (AI) and Natural Language Processing (NLP). While this goal is challenging, research progress in dialogue systems has been made throughout the years. One such progress is in the line of Task-Oriented Dialogue (ToD) systems that aim to assist human users in accomplishing a specific task. This chapter lays out the context of the study of this thesis, namely Natural Language Understanding (NLU) or Spoken Language Understanding (SLU)[1] in ToD systems and its challenges, sets up the research objective, and highlights the contributions and the research roadmap carried out in this thesis.

1.1 Context

A ToD system is an intelligent system that interacts with human users through a conversation (via voice or text) to complete a particular task (Young et al., 2010; hao Su et al., 2018). Tasks are typically well-constrained and practical *functionalities* such as booking a flight, checking the weather in a particular location, playing a song, or setting up an appointment (Figure 1.1). ToD systems have been around in the research community for a while (Raux et al., 2005; Bohus and Rudnicky, 2009; Wen et al., 2017) and also attract

[1]Some literatures refer to NLU as SLU.

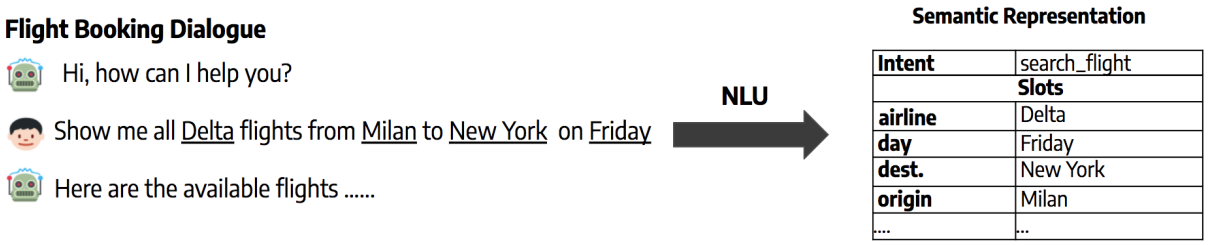


Figure 1.1: Given an input utterance, NLU parses it into a semantic representation that contains the intention of the user (intent) and relevant information (slots) corresponding to the intent.

significant interest in the industry, especially since the launch of Siri[2], Google Assistant[3], Alexa[4], and Cortana[5] (McTear, 2020).

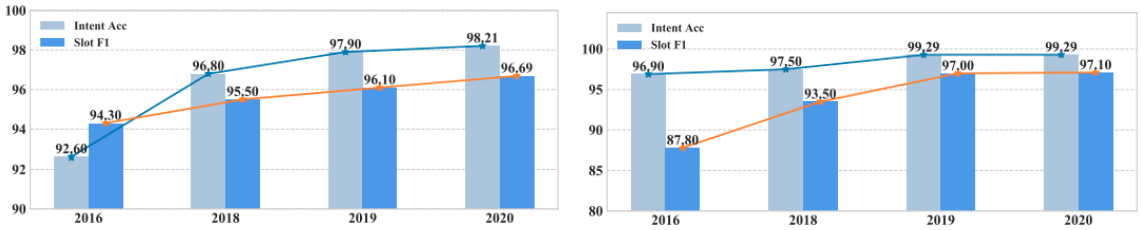
It is essential that during its interaction with the user, the ToD system needs to have the ability to understand what the user wants, decide what action needs to be performed, and generate a response back to the user. These abilities are typically realized in a pipeline framework (Young, 2000). There are several components in the pipeline of a ToD system, namely Natural Language Understanding (NLU), Dialogue Manager (DM), and Natural Language Generation (NLG). When the input utterance is from speech, an Automatic Speech Recognition (ASR) component is needed to convert the speech input into text. The NLU component is responsible for parsing the utterance into a particular semantic representation. The DM then uses the semantic representation to update the current state of the dialogue and determine the next action to be carried out, e.g., querying an external knowledge base, asking for the required information from the user. Finally, given the next action, the NLG generates the response back to the user. This thesis focuses on the NLU component of a ToD system.

In the context of ToD systems, NLU deals with parsing a user utterance into a semantic representation. The representation is often modeled as a semantic frame (Tur and De Mori, 2011), which captures the intention of the user and relevant information mentioned in the utterance that corresponds to the intent (Figure 1.1). This process involves two tasks, namely *intent classification* and *slot filling*. For example, given the

[2]https://www.apple.com/siri/.
[3]https://assistant.google.com/ .
[4]https://developer.amazon.com/en-US/alexa .
[5]https://www.microsoft.com/en-us/cortana .

utterance "*Show me all Delta flights from Milan to New York on Friday*", the intent is SEARCH_FLIGHT and the corresponding slots are "*Delta*" (AIRLINE), "*Milan*" (ORIGIN), "*New York*"(DESTINATION), and "*Friday*"(DAY). These intents and slots are *pre-defined* and *domain-specific*.

1.2 Motivation and Research Problem



(a) ATIS (b) SNIPS

Figure 1.2: Recent performance trend for NLU tasks: intent accuracy and slot F1 on ATIS and SNIPS datasets when sizeable training data is available for each domain. Charts from Qin et al. (2021).

Although existing neural network-based approaches (Liu and Lane, 2017; Wang et al., 2018a; Goo et al., 2018) on NLU have shown substantial progress with state-of-the-art performance (Figure 1.2) on standard benchmark datasets such as ATIS (Price, 1990) and SNIPS (Coucke et al., 2018), these approaches are typically trained on large labeled data[6] (*data hungry*). In addition to that, in terms of language, most studies are defined on the availability of the training data, which is centered around English datasets such as ATIS (Price, 1990), MIT corpus (Liu et al., 2013), and SNIPS (Coucke et al., 2018).

With the widespread use of commercial conversational agents such as Google Assistant, Amazon Alexa, Microsoft Cortana, and Apple Siri, these systems are continuously updated with more functionalities and language coverage to reach a broader audience. For instance, it is reported in 2018 that Alexa devices already have around 70,000 functionalities[7] and support eight languages as of April 2021[8].

[6] In the ATIS and SNIPS datasets, the full training data size is 4K and 13K, respectively.
[7] ZDNet.com. Accessed June 2021
[8] Summa Linguae. Accessed June 2021

Typically functionalities in task-oriented dialogue systems are grouped into domains. While the notion of a domain in general NLP usually refers to text variation in multiple dimensions such as topics, genre, style, medium, etc., on a given dataset (Plank, 2011; Ramponi and Plank, 2020), the notion of a domain in ToDs is relatively narrower. In the context of NLU in ToDs, a domain usually defines the required intentions that need to be extracted from the user utterances so that the system can perform its task effectively (Tur and De Mori, 2011). Typically a domain is represented by a collection of predefined intents and slots that define the domain's target semantic. For example, in a flight domain, as we have seen in Figure 1.1, we can have the utterance "*Show me all Delta flights from Milan to New York on Friday*", in which the intent is SEARCH_FLIGHT and the corresponding slots are "*Delta*" (AIRLINE), "*Milan*" (ORIGIN), "*New York*"(DESTINATION), and "*Friday*"(DAY). On another domain, for example, a restaurant booking domain, we can have the utterance "*Find me a Mexican restaurant*" in which the intent is SEARCH_RESTAURANT and the corresponding slot is "*Mexican*" (CUISINE).

As *domains* and *languages* in the system are expanding, the NLU engine should be able to adapt and recognize new intent and slots types. The straightforward solution is achieved through a standard *supervised learning* by training a separate NLU model with labeled data from each domain and language pair. However, this solution does not scale well, as defining the slots and intents types, collecting and annotating a large amount of domain-specific training data every time a new domain and language is introduced can be time-consuming. Thus, *developing an NLU model that generalizes well across domains and languages with less labeled data is necessary and remains a challenging problem.*

1.3 Research Aim and Scope

In response to the problem description in the previous subsection, this thesis aims to investigate methods to improve the model performance on NLU tasks in *low- resource* scenarios, i.e., limited labeled data on a particular domain or language.

Methods. Building on recent research progress on *transfer learning* (Pan and Yang, 2010; Ruder, 2019) and *data augmentation* methods, in this thesis we investigate their applicability on NLU tasks in ToD systems when limited labeled data is available. Trans-

fer learning essentially leverages knowledge from a particular source dataset or model to the target settings. On the other hand, data augmentation aims to produce additional training data automatically to improve model performance.

In the context of NLP research, the idea of transfer learning has been around for quite a while. It is prevalent with the ubiquitous use of pre-trained word embeddings such as Word2Vec(Mikolov et al., 2013), Glove (Pennington et al., 2014), until Elmo (Peters et al., 2018) and BERT (Devlin et al., 2019) which provides contextual embeddings and pre-trained models which can be fine-tuned to many NLP tasks. For data augmentation, although it has been well received in ML research, in particular for computer vision tasks, it is relatively underexplored in NLP partly due to the discrete representation of the input space (Feng et al., 2021). Nevertheless, there has been an increasing interest in studying data augmentation in NLP, from basic operation like swapping and word replacement (Wang and Yang, 2015; Wei and Zou, 2019b), to the more complex sentence generation using pre-trained NLG models (Quteineh et al., 2020).

Low-Resource Scenario. In the NLP literature, a low-resource scenario typically depends on a particular assumption of resource availability that is used in the study. Typically these assumptions fall into several dimensions: the availability of the task-specific labels, the availability of unlabeled language or domain-specific text, and the availability of auxiliary data (Hedderich et al., 2021). This thesis investigates low-resource scenarios where there is a lack of task-specific labeled data in the target domain (or language) for training the NLU model. Regarding the definition of "*how low is low-resource*", different settings have been used in the context of NLU. Most studies on English NLU (Chen et al., 2016; Jaech et al., 2016; Hou et al., 2018a; Peng et al., 2020) use a small percentage of the original training data e.g. 10% to simulate low-resource settings when a new domain is added to the system. Similarly, recent studies (Hou et al., 2020; Henderson and Vulić, 2021) evaluate their methods on few-shots settings, in which only one or two examples are available for training (1-shot, 2-shot). While in cross-lingual NLU, where non-English languages are used, the low-resource scenario can be framed as zero-shot settings (Upadhyay et al., 2018; Schuster et al., 2019a; van der Goot et al., 2021), in which only a "high-resource" language, e.g., English labeled dataset is available for training and evaluation is performed on several non-English languages. As for this thesis, the low-resource scenario is achieved by using a small percentage of the original training data or zero-shot setup in

cross-lingual NLU.

1.4 Contributions

The following items summarize each of the contribution of the work in this thesis. Throughout the thesis we refer each of the contribution as **C1 - C4**.

C1. Leveraging Non-Conversational Text As a Source of Transfer.
Transfer learning approaches for low-resource NLU leverage relatively large data in the source domain to help a task in a target domain through adaptation methods. Most of these methods assume the availability of labeled domain-specific slot filling datasets on the source domain. Little effort has been carried out in exploiting cheaper alternatives on the source side that can be useful, especially when no conversational slot filling is not available yet (*cold start*). To this end, we investigate the potential of leveraging *non-conversational* data that is labeled with a task *similar* to slot filling. We show that using Named Entity Recognition (NER) and Semantic Tagging as auxiliary tasks in a multi-task learning framework consistently improve model performance on the slot filling task in low-resource settings. Furthermore, we investigate the benefit of applying data selection prior to multi-task learning and find that, in our settings, it does not boost performance further.

C2. Generating Additional Labeled Data via Lightweight Augmentation.
Recent data augmentation approaches for low-resource NLU mostly rely on deep learning based models. We consider these approaches as *heavy weight* as we need to train a separate neural model for augmentation and also typically consist of several stages such as pre-training, ranking, and filtering of generated data. We propose Lightweight Augmentation that does not require model training which consists of a set of simple yet effective text span and sentence level augmentation operations. The augmentations include meaning-preserving slot value substitution and syntactic manipulation. In terms of performance, Lightweight Augmentation is competitive with state-of-the-art deep learning based augmentation models. Lightweight Augmentation is also beneficial for fine-tuned large pre-trained models such as BERT, ALBERT, and RoBERTa, which suggests that combining transfer learning and data augmentation is additive to NLU performance. Furthermore, we also show that Lightweight Augmentation is effective

for six non-English languages.

C3. Continued Pre-training for Zero-shot Cross Lingual NLU. Recent studies have shown that adding a second stage of pre-training of a pre-trained language model can improve performance in certain settings. However, prior studies are limited to English and mostly on text classification tasks. To this end, we systematically study the effectiveness of continued domain adaptive pre-training of a massive multilingual transformer (MMT) model on intermediate English unlabeled spoken language data for zero-shot cross-lingual tasks in eight languages. In terms of performance, we observe that continued pre-training is more effective for Latin script languages and slot filling benefits more from continued pre-training than intent classification. We observe that domain similarity between the unlabeled data and English fine-tune data is important, and using different languages between pre-training and fine-tuning can hamper performance.

C4. Survey of Recent Neural Methods for Slot Filling and Intent Classification. Since the adoption of neural approaches in NLP, there has been a lot of development of applying different kinds of neural architectures to SF and IC tasks. We conduct a survey to give a guided map to existing neural approaches for NLU tasks. Our survey includes a broad overview of three neural architectures: *independent models*, which models slot filling and intent classification tasks separately, *joint models* which exploits the synergy of both tasks simultaneously, and *transfer learning* models that scale NLU models to new domains.

1.5 Thesis Outline

This thesis consists of six further chapters. Chapter 2 introduces concepts about neural networks and common neural architectures that are often used in NLP tasks. We describe aspects of human conversation which illustrate the complexity and challenges of building dialogue systems. We describe two types of dialogue systems: chat-oriented and task-oriented dialogue systems and their rule-based and data-driven approaches.

In Chapter 3 we describe in more detail about the NLU task in the context of ToD systems. The content of this chapter corresponds to the contribution **C4** (§1.4), which includes a literature review on the recent progress of applying neural methods for slot filling and intent classification tasks, and state of the art approaches for low-resource

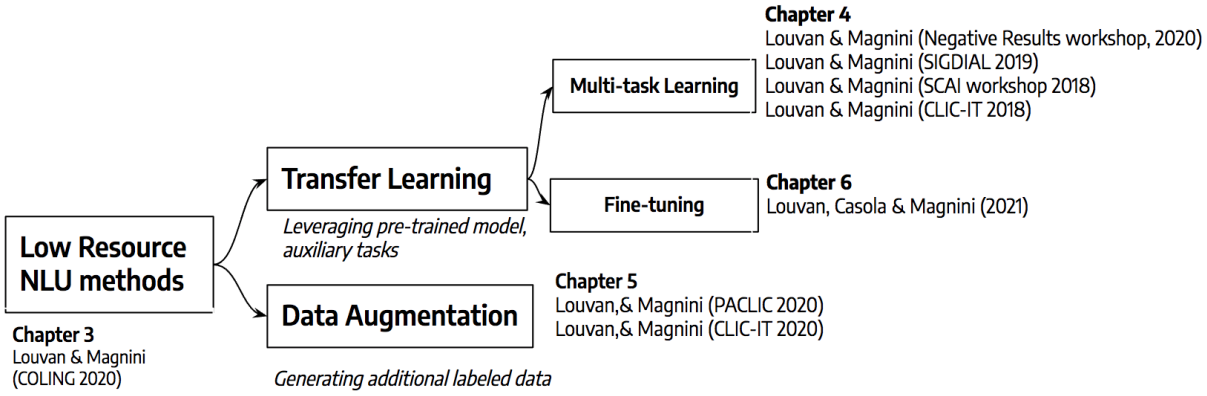


Figure 1.3: Topics for each chapter and its corresponding publications

NLU. We provide a comparison of each approach and conclude by drawing a connection from existing studies and contributions of this thesis. The content of this chapter is partially taken from Louvan and Magnini (2020b).

In Chapter 4, our study (Louvan and Magnini, 2019) examines a specific setting in which no task-oriented dataset is available as auxiliary data. We study the potential of leveraging a more general source from *non-conversational text* which has similar task characteristics with NLU (contribution **C1** in §1.4) via multi-task learning (MTL). Furthermore, as a follow up work (Louvan and Magnini, 2020a), we also examine the potential of performing data selection on the auxiliary data before performing MTL.

In Chapter 5, we turn into a relatively emerging method in in NLP, data augmentation, and its combination with state of the art transfer learning methods (contribution **C2** in §1.4). We propose a set of simple and effective augmentation methods (LIGHWEIGHT AUGMENTATION) (Louvan and Magnini, 2020d) to automatically generate additional labeled data. We also evaluate the applicability of LIGHWEIGHT AUGMENTATION on several domains and non-English languages (Louvan and Magnini, 2020c).

In Chapter 6, motivated by the recent availability of multilingual datasets and progress on pre-trained multilingual models, we examine the effectiveness of additional pre-training on intermediate *unlabeled spoken language data* for zero-shot cross-lingual SLU (contribution **C3** in §1.4) on eight languages. We also analyze the important factors to consider when performing continued pre-training in the context of zero-shot cross-lingual NLU. Finally, Chapter 7 draws conclusions from the thesis and highlights possible future works.

1.6 Publications

The following list of publications are published work and included in the discussion of this thesis:

1. **Samuel Louvan** and Bernardo Magnini. From General to Specific : Leveraging Named Entity Recognition for Slot Filling in Conversational Language Understanding. *In Proceedings of the Fifth Italian Conference on Computational Linguistics* (CLiC-it). 2018

2. **Samuel Louvan** and Bernardo Magnini. Exploring named entity recognition as an auxiliary task for slot filling in conversational language understanding. *In Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd International Workshop on Search-Oriented Conversational AI*, Brussels, Belgium. 2018

3. **Samuel Louvan** and Bernardo Magnini. Leveraging non-conversational tasks for low-resource slot filling: Does it help? *In Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*. 2019

4. **Samuel Louvan** and Bernardo Magnini. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. *In Proceedings of the 28th International Conference on Computational Linguistics (COLING)*. 2020

5. **Samuel Louvan** and Bernardo Magnini. Simple is better! Lightweight data augmentation for low-resource slot filling and intent classification. *In Proceedings of the 34th Pacific Asia Conference on Language, Information and Computation* (PACLIC). 2020

6. **Samuel Louvan** and Bernardo Magnini. Simple data augmentation for multilingual NLU in task oriented dialogue systems. *In Proceedings of the Seventh Italian Conference on Computational Linguistics* (CLiC-it) . 2020

7. **Samuel Louvan** and Bernardo Magnini. How far can we go with data selection? a case study on semantic sequence tagging tasks. *In Proceedings of the First Workshop on Insights from Negative Results in NLP*. 2020

8. **Samuel Louvan**, Silvia Casola, and Bernardo Magnini. Investigating Continued pretraining for Zero-Shot Cross-Lingual Spoken Language Understanding. *In Proceedings of the Eight Italian Conference on Computational Linguistics* (CLiC-it). 2021

9. Bernardo Magnini and **Samuel Louvan**. Understanding Dialogue for Human Communication. *Handbook of Cognitive Mathematics, Springer Nature*. 2021

Chapter 2

Background

This chapter presents concepts that are used throughout the thesis. Section 2.1 summarizes neural network architectures that are commonly used in data-driven approaches in NLP. Section 2.2 reviews concepts related to dialogue systems including, aspects in human conversation and two main categories of dialogue systems: chat-oriented dialogue and task-oriented dialogue (ToD) systems that include its rule-based and data-driven approaches. In the context of this thesis, the focus is on ToD systems.

2.1 Neural Networks

In recent years, with the rapid advances of neural network approaches (*deep learning*), faster computation with Graphical Processing Unit (GPU), *neural based models* have become the default approach to solve many NLP tasks (Manning, 2015). This section provides a brief overview of several concepts on neural networks used throughout this thesis, namely Feed-Forward Network, Recurrent Neural Network, and Self-Attention Network (Transformers).

2.1.1 Feed Forward Network

The feed-forward network (FFN) or often called Multi-Layer Perceptron (MLP) (Rosenblatt, 1957) is the simplest building block of a neural network model. It is called feed-

forward as information flows in one direction. The FFN consists of several layers: *input representation* layer, *hidden* layer, and the *output* layer. The number of layers is flexible, and the model is often referred to as a *deep* model when many layers are used. The input representation of an FFN is a *vector* representation of the text input. Nowadays, the standard representation of the text input (e.g., words) is an embedding that is a dense vector representation of words. The embeddings can be learned from scratch or can be pre-trained on large unlabeled data using methods such as Word2Vec (Mikolov et al., 2011) and Glove (Pennington et al., 2014). The next hidden layer then transforms the input into another representation (*hidden state representation*). Finally, the output layer uses this hidden state representation to compute the final output.

Essentially, transforming the representation in each hidden layer of a FFN is performed by applying a matrix multiplication and non-linear function to the previous representation (Figure 2.1). For example, given an input vector $\mathbf{x}$, the next layer has a parameter in a matrix $\mathbf{W_1}$ and a bias vector $\mathbf{b_1}$. To compute the hidden state vector representation, $\mathbf{h}$, we compute $\mathbf{h} = g(\mathbf{W_1 x} + \mathbf{b_1})$, where g is a non-linear activation function such as ReLu, tanh. The output layer then takes $\mathbf{h}$ to compute the final output vector, $\mathbf{z}$, where $\mathbf{z} = g(\mathbf{W_2 h}) + \mathbf{b}_2$. In many cases, $\mathbf{z}$, is used for classification decision and we want to turn the real valued numbers in $\mathbf{z}$ into a probabilistic distribution $\mathbf{y}$ of the output classes via a softmax function. Figure 2.1 describes the standard operation in a FFN.

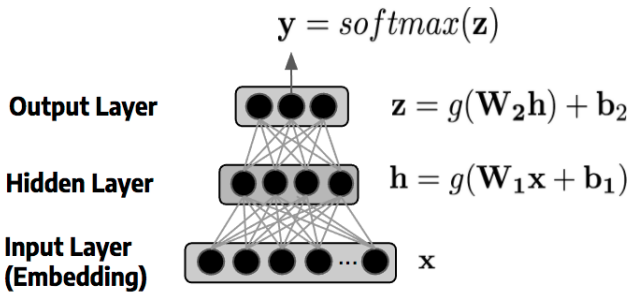


Figure 2.1: A feed forward neural network. Each unit in each layer is fully connected to the other units in the subsequent layer.

2.1.2 Recurrent Neural Network

In language processing, an input is typically processed in *sequence*, e.g., a paragraph is processed sentence by sentence and word after word in a sentence. In addition to that,

we need to consider the temporal context information as each word may depend on other words that come before or after the current word. The use of FFN in NLP has several shortcomings when used for language processing. FFN uses a fixed-size input which can be impractical as in language processing the input such, i.e., a sentence can be a *variable-length*. FFN also does not naturally capture the temporal context of the sequence as it represents an input at once in this fixed-size input. Recurrent Neural Network (RNN) has been used to address these challenges.

The Elman network (Elman, 1990) is one of the simplest examples of a RNN. Similar to FFN, a RNN processes an input $\mathbf{x}_t$ multiplied with a parameter matrix and applying a non-linear activation function to get the hidden state $\mathbf{h}_t$ which will be used by the output layer to compute the final output $\mathbf{y}_t$. In contrast to FFN, the RNN processes one input at a time and the hidden state from the previous time step, $\mathbf{h}_{t-1}$, is incorporated in computing the hidden state representation in the current time step, $\mathbf{h}_t$. The previous step's hidden state captures the context and memory from the beginning of the input sequence. Hence, RNNs can handle variable-length input and capture temporal aspects in the sequence. Figure 2.2 illustrates the structure of an RNN when processing an input sequence.

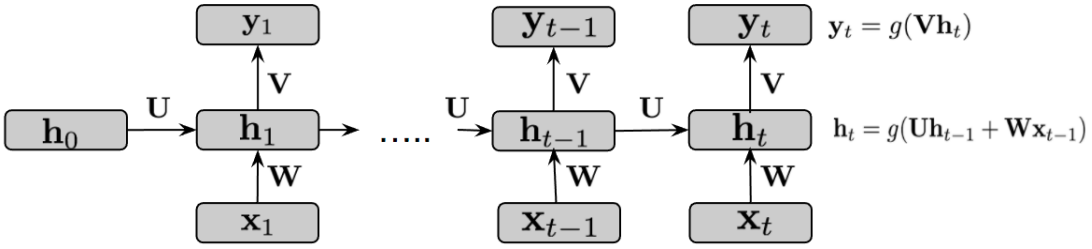


Figure 2.2: An unrolled recurrent neural network when processing an input sequence. The parameter matrices $\mathbf{U}, \mathbf{V}, \mathbf{W}$ are shared across time steps.

RNN Variants. One known problem with RNNs is the vanishing-gradient problem as the input sequence gets longer, makes it difficult to capture long-dependencies. More complex activation unit have been proposed to alleviate this problem such as Gated Recurrent Unit (GRU) (Cho et al., 2014) and Long Short-Term Memory (LSTM) network (Hochreiter and Schmidhuber, 1997). The GRU has mechanisms to decide whether the information from the previous hidden state needs to be included at all in the current time step (*reset gate*) and how much information from the previous hidden state needs to be included in

the current hidden state (*update gate*). The goal of LSTM is similar to GRU but with more complex activation units.

RNN building blocks. There have been many variations in composing a neural model using an RNN as the building block. As in FFN, RNN can be stacked for several layers. In addition to that, instead of one direction left to right sequence processing, it has been shown that a bi-directional RNN (Schuster and Paliwal, 1997) learn better context information and perform better than the vanilla RNN in many NLP tasks such as Part-of-Speech tagging (Plank et al., 2016) and Named Entity Recognition (Chiu and Nichols, 2016).

2.1.3 Self-Attention Network: Transformer

The transformer model is introduced by Vaswani et al. (2017) which is initially applied to Machine Translation (MT) tasks. Compared to RNN, which processes inputs sequentially in nature, Transformer allows better parallelization when processing a sequence of input. Instead of processing one input at a time like RNN, Transformer is more similar to FFN because it processes elements in the sequence at once. The Transformer typically consists of several layers of a *transformer block* which include a self-attention layer, a feed-forward layer, and a residual connection, among other details. Transformer uses *self-attention* mechanism where each token in the input compares its representation with the rest of the tokens in the same input to learn better context information and relation between tokens. This self-attention computation is performed in several layers, e.g., six layers. The entire architecture from the original Transformer is shown in Figure 2.3.

We briefly describe some relevant concepts in a single transformer block:

Self-Attention. Instead of using the raw input embedding, the self-attention is computed using the notion of a query (q), key (k), and value (v) attention to capture different roles of the input embedding. The query is used when a token is compared to all other tokens in the same input. The key is used to respond to a query, and the value is used to compute the attention output. As each input can be calculated independently, the computation can take advantage of efficient matrix multiplication:

$$SelfAttention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = softmax(\frac{\mathbf{Q}\mathbf{K}^\mathbf{T}}{\sqrt{d_k}})\mathbf{V} \tag{2.1}$$

Multi-head attention. In a sentence, different words can have a relation with the other

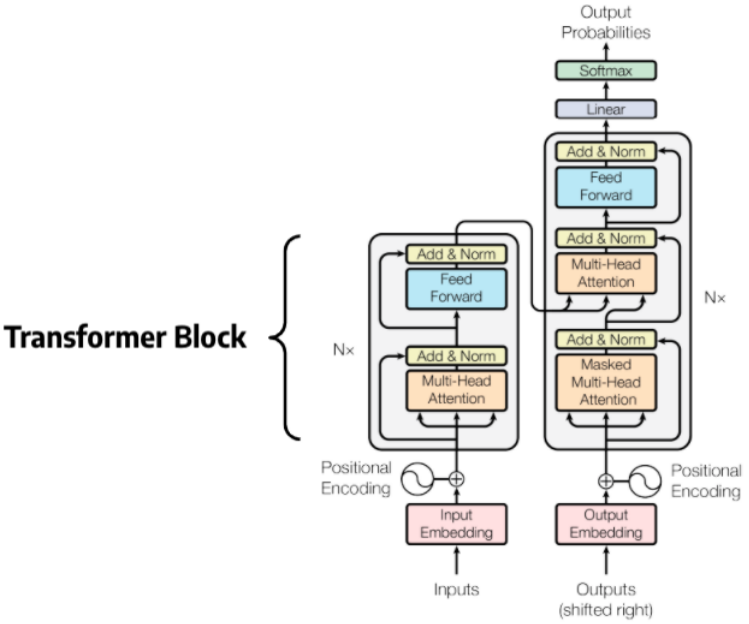


Figure 2.3: The Transformer model from Vaswani et al. (2017)

words on various aspects, for example, syntactic, semantic, and discourse relationships. Learning these different relations are the motivation of the *multi-head attention* which is a set of self-attention layers.

Positional Embedding. In an RNN, information of the order of the inputs is inherently part of the model. However, for Transformer, no information indicates the relative or absolute position of the input order. To add the word order information, the input embedding is combined with the positional embedding. The original Transformer work uses a combination of sine and cosine functions to compute a position embedding.

2.1.4 Training a Neural Network

As a neural network is an instance of supervised machine learning, it needs to be trained with example inputs $\mathbf{x}$ with its correct output $\mathbf{y}$. Given a set of N training examples, $\mathcal{D} = \{\mathbf{x}^i, \mathbf{y}^i\}_{i=1}^N$, the aim is to estimate (*learning*) a function $\mathbf{f}(\mathbf{x}; \Theta)$ that maps $\mathbf{x}$ to $\mathbf{y}$. The learning process essentially involves estimating the most optimal value of the parameters Θ. In order to do this, first we need a *loss function* that models the distance between the correct output y and the model's prediction $\hat{y}$. For example, in the context

of classification tasks, the typical loss function, $\mathbf{L}$, that is used is the *cross-entropy loss* which is defined as $\mathbf{L}(\hat{y}, y) = -\sum_{i=1}^{C} y^i \log \hat{y}_i$, over C possible classes. To find the parameters that minimize the loss, typically a particular gradient-based *optimizer* is used, for example Adam (Kingma and Ba, 2014), Adagrad (Duchi et al., 2011). The gradient of the loss is computed using the partial derivative of the loss function with respect to each parameter in the neural network layers. The gradient computation is computed using the backpropagation algorithm (Rumelhart et al., 1986).

2.2 Dialogue Systems

This subsection present concepts relation to dialogue systems. Some parts this subsection are adapted from Magnini and Louvan (2021). We describe characteristics on human dialogue (§2.2.1) that illustrates the complexity of a conversation which makes building dialogue systems challenging. We review two types of dialogue systems: chat-oriented dialogue system (§2.2.2) and task-oriented dialogue (§2.2.3) including its rule-based and data-driven approaches.

2.2.1 Characteristics of Human Dialogue

Turns & Utterance. The distinguishing characteristic of dialogues is that they appear as a sequence of *turns* ($T_1, T_2, ...$ in Table 2.1), each constituted by a speech by a single speaker, speaking without interruption by the speech of the other speaker (Traum and Heeman, 1996). Using turns, the speakers alternate their contributions to achieving the dialogue goals. At each turn, the participants make explicit their contribution through one or more *utterances* (e.g., a question, an answer) in a way that utterances keep a reasonable connection with the previous turns and with the intended goals of the dialogue.

Speech & Dialogue Act. The study of human dialogue has been greatly influenced by the theory of *speech acts*, initially introduced by Austin (1962) and then further developed by Searle (1969). The core intuition behind speech acts is that language is not only used to make statements (called *constative* speech acts), but also to perform actions that have effects in the world (called *performative* speech acts). Performative utterances do not have a true value; rather, they have *felicity conditions*, which may be appropriate or

Turn	Speaker	Utterance
T_1	Client	Can you find me a train from Cambridge to Leicester?
T_2	Agent	There are several what day and time do you want to leave or arrive?
T_3	Client	I'd like to leave on Wednesday and should arrive by 13:00
T_4	Agent	The TR2176 leaves Cambridge at 09:21 and gets into Leicester at 11:06. Will that work for you?
T_5	Client	What is the price of that train?
T_6	Agent	The price is 37.80 pounds. Would you like to go ahead a book a seat?
T_7	Client	Yes, I would like to book a seat on that train. Thank you.
T_8	Agent	Just to clarify, am I booking just one ticket or more?
T_9	Client	Yes. also find me Primavera's phone number
T_{10}	Agent	Booking was successful, the total fee is 37.79 GBP payable at the station. Reference number is : 2Z3LQ075 . What else did you need?
T_{11}	Client	Thanks. I am also looking for the Primavera. It is an attraction.
T_{12}	Agent	Yes, it's a very nice attraction.
T_{13}	Client	Can I get their phone number please?
T_{14}	Agent	Their phone number is 01223357708. Can i help with something else?
T_{15}	Client	No. That will be all. Thank you.
T_{16}	Agent	Thank you for contacting the Cambridge Town Info centre. Goodbye.

Table 2.1: Example of a human-human dialogue from the MultiWOZ dataset (Budzianowski et al., 2018) between a client and a town info agent.

not for the utterance to have the supposed consequences. For example, when we say *"I order you to go"*, then some action of going is expected to happen. In more recent years, the speech act theory has influenced the design of dialogue systems, and the notion of *dialogue act*, strictly derived from speech act, has become a central characteristic for the interpretation of user utterances. Specific inventories of dialogue acts have been proposed to be applied to a vast range of conversational situations. As an example, Stolcke et al. (2000), list 42 dialogue acts, including THANKING (e.g. T_{16} in Table 2.1 *Thank you for contacting the Cambridge Town Info centre. Goodbye.*, YES-NO-QUESTION, (e.g. T_{14} in Table 2.1 *"...Can i help with something else"*), OPINION (e.g. T_{12} in Table 2.1 *Yes, it's a very nice attraction.*), and BACKCHANNEL-QUESTION (e.g. T_8 in Table 2.1 *Just to*

clarify, am I booking just one ticket or more?).

Grounding. Dialogue requires that interlocutors agree on the topic and the entities of the conversation and acknowledge each other. The process through which the participants keep themselves aligned during the dialogue is called *grounding* (Clark and Schaefer, 1987; Clark and Brennan, 1991), and it involves acknowledging that information has been established as part of the common ground of the dialogue participants. This process has been investigated in computational linguistics in order to codify the behavior of the participants. As an example, Traum and Nakatani (2002) proposes a model based on *common ground units*, where grounding elements (e.g., repetitions, expressions of explicit agreement, like *ok*) are explicitly marked. One example of grounding through repetition is shown in T_{14} of Table 2.1 when the agent repeats the phrase "*their phone number*" from the previous utterance in T_{13} indicating that the agent understands the client.

Initiatives & Subdialogues. Conversation flow can be controlled by one participant. For example, in a sport post-match press conference, a journalist asks questions related to the match, and the athlete provides answers. So, in this case, the information flow is unidirectional, and the journalists take the conversation *initiatives*. However, in a normal human-human conversation, information flows bi-directionally where the control of the conversation can shift from one speaker to another speaker making a *mixed initiatives* (Walker and Whittaker, 1990) conversation. These mixed initiatives can happen as there can be a different sequence or subdialogues takes place, e.g., as sometimes a speaker that answers a question can also ask questions, sometimes a speaker drives the conversation to a different topic (T_9 in Table 2.1), etc.

2.2.2 Chat-Oriented Dialogue System

The ability to engage in a fluent and natural conversation with a human is often denoted as one of the requirements of a true artificial intelligence system (Turing, 1950). While this idea seems elusive and unattainable, it has attracted many researchers to work on *chatbots* which mimics human conversation behavior and able to converse on different kinds of topics (*open-domain*).

Rule-based chatbots. Early chatbots were developed to pass the Turing Test, in which a system passes the test if a human thinks that they are talking to another human. ELIZA (Weizenbaum, 1966), a chatbot developed by Weizenbaum at MIT, is often considered

as one of the pioneering examples in dialogue systems and has influenced subsequent systems such as Parry (Colby et al., 1971), TEAM (Grosz et al., 1987), and ALICE (Wallace, 2009). ELIZA relies on *pattern matching* scripts to generate a response back to the user. The programmer needs to define each of the pattern matching rules. Table 2.2 shows an example of interaction between a user and ELIZA.

Speaker	**Utterance**
USER	Men are all alike
ELIZA	In what way?
USER	They're always bugging us about something specific or other
ELIZA	Can you think of a specific example?
USER	Well, **my** boyfriend made me come here.
ELIZA	Is it important to you that your boyfriend made you come here?

Table 2.2: An example of interaction scenario between a user and ELIZA. ELIZA uses a script to simply asking back the user's statement, for example, whenever it encounters the keyword "*my*" in the user's utterance.

While ELIZA was initially created to demonstrate the superficiality of dialogue between humans and machines, it successfully deceived early users that they were actually communicating with a human.

Data-driven based chatbots. With the availability of human-human conversation data online, such as conversations on social media (e.g., Twitter, Reddit), movie subtitles, and the rapid development of applying machine learning approaches on NLP, research on dialogue systems have shifted its focus from *rule-based* to *data-driven* approaches. In a rule-based approach, a response from the system is a result of matching an utterance to a predefined template, while in a data-driven approach, rules are learned automatically from massive data to map an utterance—and its dialogue context—to an appropriate response (*response generation*). Ritter et al. (2011) formulates this response generation problem as a translation model. They exploit a Twitter conversation thread composed of a conversational stimulus and a response and use Statistical Machine Translation (SMT) tools on these conversational threads so that the SMT can generate a response given a conversational stimulus. With the rapid adoption of neural approaches on NLP, a myriad amount of neural approaches have been applied to response generation in the context of

chatbots. Many of these neural approaches (Vinyals and Le, 2015; Sordoni et al., 2015; Shang et al., 2015) use the *encoder-decoder* (SEQ2SEQ) architecture (Sutskever et al., 2014). The encoder process the utterance and its context, and the decoder generate the response word by word until the end of the utterance marker. Recently, with the advent of large pre-trained language models such as GPT-2 (Radford et al., 2019) (pre-trained on a general text), a common approach is to continue the training of the pre-trained language model to an unlabeled conversational dataset (*fine-tuning*) and use the language model to generate the response. On this line of work, some approaches perform large scale pre-training on massive conversational text such as DIALOGPT (Zhang et al., 2020) and MEENA (Adiwardana et al., 2020).

Speaker	**Utterance**
Emotion: Lonely	
USER	Hi there , don't know what to do , just broke up with my girlfriend , we were 8 years together.
MIME	Oh no ! What happened ?
Emotion: Excited	
USER	I have a huge chance to win a PS4 !
MIME	That is great ! Are you excited ?
Emotion: Disappointed	
USER	I lost my favorite hoodie somewhere . I am so upset . like how could I lose a hoodie ? It is so big , did I just leave it somewhere ?
MIME	Well that is not good , I hope you have to be careful next time?

Table 2.3: Examples of conversation between a user and MIME (Majumder et al., 2020) on different emotion scenarios.

There exist known limitations on neural chatbot systems, such as it tends to generate a response that is too general or bland by replying "OK", "I do not know" or just repeating what the user has said. In order to generate an appropriate response, the system needs to capture the semantics that involves natural language understanding (NLU) and user understanding, including recognizing entities, intent, topics, emotion, and sentiment from the user, knowledge, or commonsense reasoning (Huang et al., 2020). The NLU of a chat-oriented dialogue system is more challenging than a task-oriented dialogue system as

there is no pre-defined schema and labels as the dialogue is open-ended. Several advancements try to alleviate this problem. Some works (Zhou et al., 2018; Tuan et al., 2019) incorporated an external knowledge base to link the facts about the world—concepts, facts, relation—into the dialogue systems (*knowledge grounding*). Also, recognizing the user's emotion has been shown as another factor that influences the length of the user's engagement with the system (Shum et al., 2018). Emotionally aware dialogue systems (Zhou et al., 2020; Majumder et al., 2020) typically incorporate an emotion classifier, which is trained on a conversational dataset that is annotated with emotion labels. One such example is the EMPATHETICDIALOGUES (Rashkin et al., 2018) dataset. Table 2.3 shows an emotionally aware dialogue generation from MIME. The work from Li et al. (2016); Vijayakumar et al. (2018) incorporate *diversity factors* in the training objective to generate more diverse and appropriate responses.

In a broader context, to control text generation from a large pre-trained language model is still an open research question. As pre-trained language models are initially trained on a general corpus, often we do not have much control over the attributes of the output text such as the topic, sentiment, style, factual correctness, etc (Li et al., 2021b). Controlling these attributes are essential for applications such as dialogue system. For example, we do not want the dialogue system to generate text that contains toxic, abusive, and hatred (Gehman et al., 2020). Dathathri et al. (2020) explored an approach to steer the text generated from a pre-trained language model by plugging multiple attribute models to guide the generated text. These attribute models govern the sentiment and the topic of the generated text from the language model. Controlling text generation from a pre-trained language model is still an early and active stage of research; one can refer to survey from (Dathathri et al., 2020; Weng, 2021) for more details on recent approaches for neural controlled text generation.

2.2.3 Task-Oriented Dialogue System

TOD (often referred as *goal-oriented* dialogue) systems aim to assist users in completing a task through conversation in natural language, either in a speech or written form. In contrast to chat-oriented dialogue system, which primarily aims to maximize long-term user engagement, TOD focus on interacting with users on a domain-specific task (Gao et al., 2019), such as booking a flight or ordering food.

GUS: Hello. My name is GUS. I can help you plan a simple trip by air. Where do you want to go?
(1) *Client:* I want to go to San Diego on May 28
GUS: What time do you want to leave?
(2) *Client:* I must be in San Diego before 10 am
GUS: Would you like P.S.A. flight 102 that arrives at 9.15 am?
(3) *Client:* What is the next flight?
GUS: Air California flight 310 that leaves at 8.30 am and arrives at 10.00 am
(4) *Client:* I'll take the first one
GUS: In what name should I make a reservation, please?
(5) *Client:* Dan Bobrow
GUS: I have confirmed the following flight:
P.S.A. flight 102 on Wednesday May 28 from San Jose to San Diego
that leaves at 7.30 am and arrives at 9.15 am.
What date do you want to return on?
(6) *Client:* On Friday in the evening.
GUS: Would you like the flight that leaves at 7.45 pm?
(7) *Client:* That's fine.
GUS: I have confirmed the following flight:
P.S.A. flight 307 on Friday May 30 from San Diego to San Jose
that leaves at 7.45 pm and arrives at 9.30 pm
Thank you for calling. Goodbye.

Figure 2.4: An example dialogue taken from GUS(Bobrow et al., 1977).

One important concept in the early ToD that is still relevant until present ToD is the concept of *frame-driven* dialogue introduced in GUS (Bobrow et al., 1977). In general, a frame in GUS is a data structure that has a name and a set of *slots* associated with it. A slot also has a name and a filler (*value*). For example, in the context of travel planning in GUS, one of the frames is TRIPSPECIFICATION, which has slots such as homeport, foreign port in which the slot values are city names that define the origin and destination of a trip. These frames, slots, and their values are typically called as the *domain ontology* of the ToD. The conversation in ToD is frame-driven, as the system gets the necessary information from the user through dialogues until all required slots in the frame are filled (*slot filling*). Figure 2.4 shown an example dialogue from GUS.

Rule-based approach. In a rule-based dialogue system, the conversational flow can be pre-defined as a finite state automata (FSA) (Figure 2.5). In each state, the system asks a specific question to get the information from the user, and then based on the user's response, the system transitions to the next state. This process proceeds until the end

state is reached. During its interaction with the user, the system needs to be able to

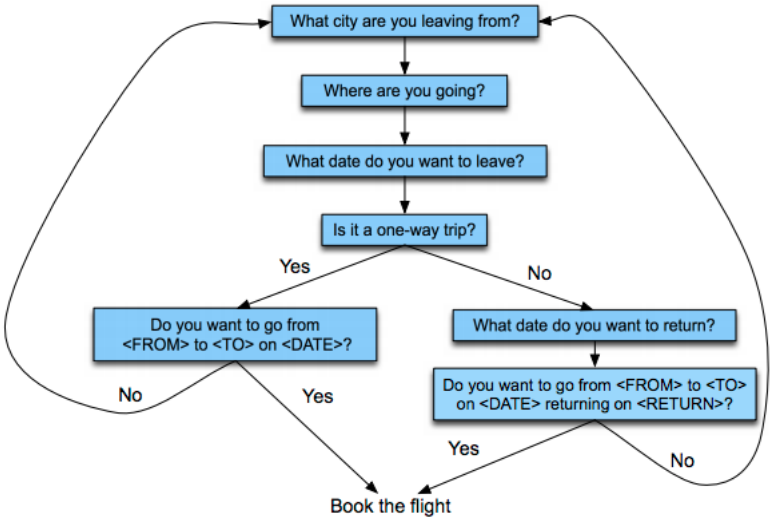


Figure 2.5: An example state transition in which there are four pre-defined slots: From, To, Date, Return in a flight booking scenario. The example is taken from Jurafsky and Martin (2009).

understand what the user has expressed. In this context, natural language understanding (NLU) is approached as detecting the slot values mentioned in the user's utterance. In a rule-based system, the system designer needs to specify semantic grammar rules for the utterance that a parser can use to extract the user's intended task and slot values.

`REQUEST`	→ show | give me
`FLIGHT`	→ flights | a flight
`ORIGIN`	→ from `CITY`
`DESTINATION`	→ to `CITY`
`CITY`	→ Milan | Trento | Barcelona

Table 2.4: Examples of semantic grammar rules

Data-driven approach (Pipeline). The arguments which motivate the move from rule-based to data-driven approaches are that rule-based approaches are more expensive to develop and not scalable when adapting to new domains, i.e., the number of rules

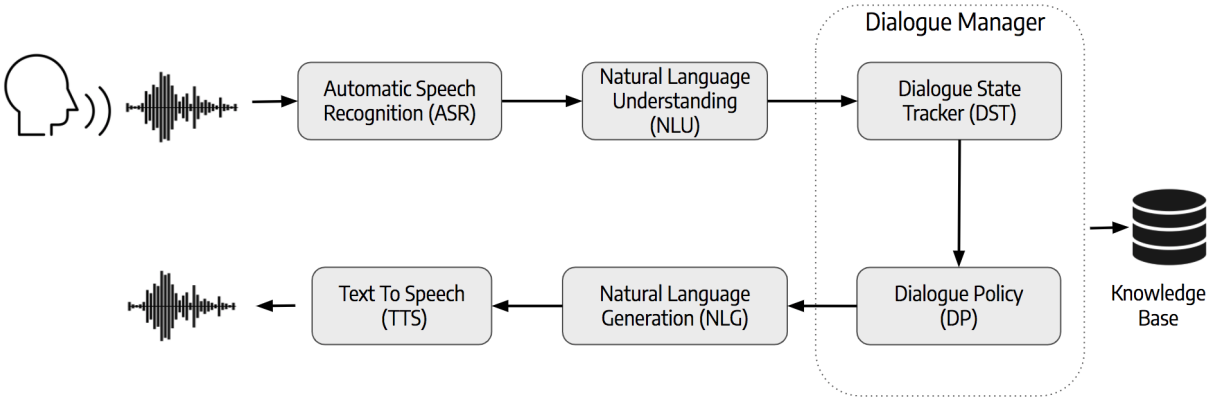


Figure 2.6: A modular architecture of a task-oriented dialogue system (Young, 2000).

may explode, which makes it difficult to maintain (He and Young, 2003; McTear, 2020). The pipeline of ToD systems typically use a *modular* architecture that consist of several components (Figure 2.6) organized in a pipeline. As shown in the overall architecture in Figure 2.6, the system consists of Natural Language Understanding (NLU), Dialogue Manager (DM), and Natural Language Generation (NLG) components. When the interaction with the user is through voice, Automatic Speech Recognition (ASR) and Text To Speech (TTS) components are needed.

NLU. In recent approaches, NLU has focused mainly on *shallow semantic parsing*, which extracts information relevant to the task that the user wants to accomplish. Typically, there are two items that are extracted from an utterance, namely the *intent*, often corresponding to the user's need (e.g., *search a flight*, *play a song*, *block a credit card*, etc.) and entities (*slots values*) that are applied to the intent, for instance *airline name, flight destination, song name, artist name*, etc. In data-driven approaches, instead of handcrafting the semantic grammar rules which can be costly to develop and hard to maintain, detecting the intent in an utterance is typically modeled as a classification problem (*intent classification*), and extracting the slots (*slot filling*) is modeled as a sequence tagging problem. In the early initiatives of the statistical-driven approach for NLU, rich features-based models, such as Support Vector Machines and Hidden Markov Model, were often applied to both intent classification and slot filling (Raymond and Riccardi, 2007; Moschitti et al., 2007). More recently, most approaches are based on neural network models (*deep learning*). Popular architectures which typically applied to these tasks are bi-LSTM + CRF (Lample et al., 2016) and by fine-tuning a pre-trained language model (Chen et al., 2019)

to the target task. We discuss further NLU models in Chapter 3.

User	I'm looking for a cheaper restaurant
	`inform(price=cheap)`
System	Sure. What kind - and where?
User	Thai food, somewhere downtown
	`inform(price=cheap, food=Thai, area=centre)`
System	The House serves cheap Thai food
User	Where is it?
	`inform(price=cheap, food=Thai, area=centre) ; request(address)`
System	The House is at 106 Regent Street

Table 2.5: An example dialogue annotated with dialogue belief state. This example is taken from Mrksic et al. (2017)

DM. The dialogue management typically involves two components, namely the Dialogue State Tracker (DST) and Dialogue Policy (DP). The DST is responsible for maintaining the *belief state* which represents the user's goal state up to the current turn of the dialogue (Henderson et al., 2014). This belief state is then used by the DP to decide which next *action* needs to be performed by the system (Su et al., 2016). In the belief state, the DST needs to track the search constraints expressed by the user (*informable slots*) and also questions that are asked by the user (*requests*) related to the search results. An example of a dialogue with informable and request slots is shown in Table 2.5. A number of approaches have been applied to DST, including using generative models such as Markov Decision Process (Williams and Young, 2007), discriminative models such as Maximum Entropy Models (Metallinou et al., 2013), linear-chain CRF (Kim and Banchs, 2014). Important to these discriminative models is the features used in the model, such as the information from NLU, the context information of dialogue history, etc. Recently, there has been a vast amount of neural approaches (Mrksic et al., 2017; Zhong et al., 2018; Lee et al., 2019; Zhou and Small, 2019; Shan et al., 2020) where less feature engineering is needed. As for DP, recently, it is dominated by reinforcement learning (RL) (Su et al., 2018) based approaches. In RL, instead of using rule-based or utterance-level annotation, it leverages the final outcome of the dialogue, whether the task is completed successfully or not. This approach, however, requires a large number of conversations, and therefore,

research has been directed to study the simulation of user interaction with the dialogue policy (Asri et al., 2016; Cao et al., 2020).

NLG. The Natural Language Generation (NLG) component receives a dialogue act from the dialogue manager as input and maps it to natural language. For example, given the system act `action=inform(cuisine=Italian,area=west, city=Firenze, price=cheap)`, NLG converts this act representation into a sentence like *There are several cheap Italian restaurants in the west area of Firenze*. Similar to other tasks, recent approaches for NLG have been dominated by neural-based approaches in particular the encoder-decoder model (Wen et al., 2015a,b; Kennard et al., 2017; Kale and Rastogi, 2020).

Data-driven approach (End to End). One known problem in pipeline methods is an error that happens in a component that can propagate to the next components along the pipeline. Therefore, there have been research effort on end-to-end (E2E) approaches of task-oriented dialogue system (Rojas-Barahona et al., 2017; Madotto et al., 2018; Wu et al., 2019; Qin et al., 2020b). In E2E approaches, no individual components, e.g., NLU, NLG, DST, are present. The input of E2E models is only the raw dialogue history, the knowledge base (i.e., domain ontology), and the model needs to output the response to the user directly. So, in this case, all information that exists in pipeline methods, such as intent, slots, dialogue state, and dialogue policy, is in the latent space. While it may reduce the error cascading, it is still challenging to capture all this information and its correlation with each other, and typically E2E models need a large amount of conversational data. In addition to that, as all information that exist in the pipeline methods is in the latent space, it could make E2E model harder to fix when error occurs.

Chapter 3

Natural Language Understanding in ToD Systems

This chapter serves as an intermediate chapter prior to next chapters that details the contribution **C1**(Leveraging Non-Conversational Text As a Source of Transfer), **C2** (Generating Additional Labeled Data via Lightweight Augmentation), and **C3** (Continued Pre-training for Zero-shot Cross Lingual NLU) in this thesis. The content of this chapter is partially derived from our survey paper Louvan and Magnini (2020b) which corresponds to the contribution **C4** (Survey of Recent Neural Methods for Slot Filling and Intent Classification).

This chapter introduces background on NLU in the context of ToD systems, presents the NLU definition as intent classification (IC) and slot filling (SF) tasks, describes its evaluation metrics, and presents recent approaches on neural NLU models. After that, we present and summarize the state of the art of low-resource NLU methods, the focus of this thesis, to scale NLU models to new domains (§3.6.1) and languages (§3.6.2) with limited labeled data. At the end of each subsection (§3.6.1, §3.6.2) we provide comparison and takeaways of existing low-resource NLU methods. We conclude this chapter with an overall summary and drawing connection from existing studies that motivate our work in **C1**, **C2**, **C3**.

3.1 Introduction

We generalize several recent approaches assuming that the output of the NLU process is a partially filled semantic frame (Wang et al., 2005; Tur and De Mori, 2011), corresponding to the intent of the user in a certain portion of the dialogue, with a number of slot-value pairs that need to be filled to accomplish the intent. The notion of *intent* originates from the idea that utterances can be assigned to a small set of *dialogue acts* (Stolcke et al., 2000), and it is now largely adopted to identify a task or action that the system can execute in a certain domain. *Slot-value pairs*, on the other end, represent the domain of the dialogue, and have been actually implemented either as an ontology (Bellegarda, 2013), possibly with reasoning services (e.g. checking the constraints over slot values) or simply trough a list of entity types that the system needs to identify during the dialogue.

Intents may correspond either to specific needs of the user (e.g. blocking a credit card, transferring money, etc.), or to general needs (e.g. asking for clarification, thanking, etc.). Slots are defined for each intent: for instance, to block a credit card it is relevant to know the name of the owner and the number of the card. Values for the slots are collected through the dialogue, and can be expressed by the user either in a single turn or in several turns. At each user turn in the dialogue the NLU component has to determine the intent of the user utterance (*intent classification*) and has to detect the slot-value pairs referred in the particular turn (*slot filling*).

In Section 2.2.3, we described a task-oriented system as a pipeline of components, saying that SF and IC are core tasks at the NLU level. Particularly, IC consists of classifying an utterance with a set of pre-defined intents (Ravuri and Stolcke, 2015), while SF is defined as a sequence tagging problem (Raymond and Riccardi, 2007; Mesnil et al., 2013), where each token of the utterance has to be tagged with a slot label. In this scenario, training data for SF typically consist of single utterances in a dialogue where tokens are annotated with a pre-defined set of slot names, and slot values correspond to arbitrary sequences of tokens. In this perspective, it is worth mentioning a research line on dialogue state tracking (DST) (Henderson et al., 2014; Mrksic et al., 2015; Budzianowski et al., 2018), where the NLU component is usually embedded into DST. What is relevant for our topic is that in this context SF is defined as a classification problem: given the current utterance and the previous dialogue history, the system has to decide whether a certain slot-value pair defined in the domain ontology is referred or not in the current utterance.

Although promising, from the NLU perspective, this research line poses constraints (e.g. all slot-value pairs have to be pre-defined in an ontology,) that limit the SF applicability. For this reason, and because NLU components are the prevalent solution in current task-oriented systems, the focus of our survey will be on SF as a sequence tagging problem, as more precisely defined in the next section.

3.2 Task Definition

We formulate SF and IC as follows. Given an input utterance $\boldsymbol{x} = (x_1, x_2, .., x_T)$, SF consists in a token-level sequence tagging, where the system has to assign a corresponding slot label $\boldsymbol{y}^{slot} = (y_1^{slot}, y_2^{slot}, .., y_T^{slot})$ to each token x_i of the utterance. On the other end, IC is defined as a classification task over utterances, where the system has to assign the correct intent label y^{intent} for the whole utterance $\boldsymbol{x}$. In general, supervised learning approaches learn a probabilistic model to estimate $p(y^{intent}, \boldsymbol{y}^{slot} | \boldsymbol{x}, \boldsymbol{\theta})$ where $\boldsymbol{\theta}$ is the parameter of the model. Table 3.1 shows an example of the expected output of a model for the SF and IC tasks.

Utterance ($\boldsymbol{x}$)	I	want	to	listen	to	Hey	Jude	by	The	Beatles
Slot ($\boldsymbol{y}^{slot}$)	O	O	O	O	O	B-song	I-song	O	B-artist	I-artist
Intent (y^{intent})	play_song									

Table 3.1: Example of SF and IC output for an utterance. Slot labels are in BIO format: `B` indicates the start of a slot span, `I` the inside of a span while `O` denotes that the word does not belong to any slot.

3.3 Datasets for SF and IC

In this section, according to our task definition, we list available dialogue datasets (most of them are publicly available) where each utterance is assigned to one intent, and tokens are annotated with slot names. Most of such datasets are collections of *single turn user utterances* (i.e., not multi-turn dialogues). An example of a single-turn utterance annotation is shown in Table 3.1. The datasets covered in this section is not necessarily

exhaustive, for more details on other existing datasets one may refer to (Razumovskaia et al., 2021).

Dataset	**Lang.**	**# intent**	**# slot**	**# train/dev/test**
ATIS (Hemphill et al., 1990)	EN	18	79	4,478 / 500 / 893
MEDIA (Bonneau-Maynard et al., 2005)	FR	-	68	12,908/1,259/3,005
SNIPS-NLU (Coucke et al., 2018)	EN	7	39	13,084 / 700 / 700
	IT	7	39	5,742 /700 / 700
Facebook Multilingual (Schuster et al., 2019a)	EN	12	11	30,521 / 4,181 / 8,621
	TH	12	11	3,617 / 1,983 / 3,043
	ES	12	11	2,156 / 1,235 / 1,692
MIT Restaurant (Liu et al., 2013)	EN	-	8	6,128 / 1,532 / 1,521
MIT Movie (Liu et al., 2013)	EN	-	12	7,820 / 1,955 / 2,443

Table 3.2: Single-turn datasets statistics. The acronyms EN, FR, IT, TH, ES correspond to the language used in the dataset, namely English, French, Italian, Thai, and Spanish.

The ATIS (Airline Travel Information System) dataset (Hemphill et al., 1990) is the most widely used single-turn dataset for NLU benchmarking. The total number of utterances is around 5K utterances that consist of queries related to the airline travel domain, such as searching for a flight, asking for flight fare, etc. While it has a relatively large number slot and intent labels, the distribution is quite skewed; more than 70% of the intent is a flight search. The slots are dominated by a slot that expresses location names such as FromLocation and ToLocation. The MEDIA dataset (Bonneau-Maynard et al., 2005) is constructed by simulating the conversation between a tourist and a hotel representative in the French language. Compared to ATIS, the MEDIA corpus size is around three times larger; however, MEDIA is only annotated with slot labels. The slots are related to hotel booking scenarios such as the number of people, date, hotel facility,

relative distance, etc. The MIT corpus (Liu et al., 2013) is constructed through a crowdsourcing platform where crowd workers are hired to create natural language queries in English and annotate the slot label in the queries. The MIT corpus covers two domains, namely movie and restaurant, in which the utterances are related to finding information of a particular movie or actor, searching or booking a restaurant with a particular distance and cuisine criteria. The SNIPS dataset (Coucke et al., 2018) was collected by crowdsourcing through the SNIPS voice platform. Intents include requests to a digital assistant to complete various tasks, such as asking the weather, playing a song, book a restaurant, asking for a movie schedule, etc. SNIPS is now often used as a benchmark for NLU evaluations.

While most datasets are available in English, recently there has been growing interest in expanding slot filling and intent classification datasets to non-English languages. The original ATIS dataset has been derived into several languages, namely Hindi, Turkish (Upadhyay et al., 2018), and Indonesian (Susanto and Lu, 2017). The MultiATIS++ dataset from Xu et al. (2020) expands the ATIS dataset to more languages, namely Spanish, Portuguese, German, French, Chinese, and Japanese. The work from (Bellomaria et al., 2019) introduces the Italian version of the original SNIPS dataset. The Facebook multi-lingual dataset (Schuster et al., 2019a), introduced a dataset on Thai and Spanish languages across three domains namely weather, alarm, and reminder. The detailed statistics of each dataset are listed in Table 3.2.

3.4 Evaluation Metrics

For the IC task, evaluation is performed on the utterance level. The typical evaluation metric for IC is *accuracy*, calculated as the number of the correct predictions made by the model divided by the total number of predictions.

As for SF, the evaluation is performed on the entity level. The common metrics used is the metric introduced in CoNLL-2003 shared task (Tjong Kim Sang and De Meulder, 2003) to evaluate Named Entity Recognition (NER) by computing the F1 score, the harmonic mean score between precision and recall. Precision (P) is the percentage of slot predictions from the model which are correct, while recall (R) is the percentage of slots in the corpus that are found by the model. These metrics are computed in terms of true

positive (TP), false positive (FP), and false negative (FN), which are defined as follows:

$$P = \frac{TP}{TP + FP} \quad R = \frac{TP}{TP + FN} \quad F1 = \frac{2 \times P \times R}{P + R} \tag{3.1}$$

A slot prediction is considered *correct* when an *exact* match is found (Tjong Kim Sang and De Meulder, 2003). As the slot is annotated in BIO format (Tjong Kim Sang and Buchholz, 2000) to mark the chunk boundary of the slot (see Table 3.1), a correct prediction is only counted when the model can predict the correct slot label on the correct boundary. Consequently, the exact match metrics does not reward cases when the model predict correct slot label but get the incorrect slot boundary (*partial match*).

Token	**Gold Standard**	**Prediction**
Show	O	O
me	O	O
the	O	O
cheapest	B-relative_fare	O
American	B-airline_name	B-airline_name
Airlines	I-airline_name	I-airline_name
flight	O	O
from	O	O
Milan	B-origin_city	B-destination_city
to	O	O
New	B-destination_city	B-destination_city
York	I-destination_city	O
on	O	O
Friday	B-day	O

Table 3.3: Example of gold standard annotation of slots in an utterance and the system prediction.

Table 3.3 shows an example of system predictions and Table 3.4 summarizes the calculation of TP, FP, and FN. Applying Equation 3.1, the precision, recall, and F1 are 33%, 20%, and 25%, respectively.

Slot	TP	FP	FN
relative_fare	0	0	1
airline_name	1	0	0
origin_city	0	0	1
destination_city	0	2	1
day	0	0	1
Total	1	2	4

Table 3.4: Calculation of TP, FP, FN for the example in Table 3.3

3.5 NLU Models

In the following sections, we outline the main models that have been proposed for SF and IC, and categorize the models into two main groups, namely *independent models* (§3.5.1) and *joint models* (§3.5.2).

3.5.1 Independent Models

Independent models train each task *separately* and recent neural models typically use RNN as the building block for SF and IC. At each time step t, the encoder transforms the word representation x_t to the hidden state h_t. For SF, the output layer predicts the slot label y_t^{slot} condition on h_t. For IC, typically the last hidden state h_T is used to predict the intent label $y^{intent}$of the utterance $\boldsymbol{x}$. Note that, for independent approaches, the models for SF and IC are trained separately. Most neural models for SF and IC generally consist of several layers, namely an *input layer*, one or more *encoder layer*, and an *output layer*. Consequently, the main differences between models are in the specifics of these layers. The most common dataset used for evaluating independent models is ATIS.

In the *input layer* of neural models each word is mapped into embeddings. Mesnil et al. (2013) compared several embeddings, namely pre-trained SENNA (Collobert et al., 2011), RNN Language Model (RNNLM) (Mikolov et al., 2011), and random embeddings. SENNA gives the best result compared to other embeddings, and, typically, further fine-tuning word embeddings improves performance. Yao et al. (2013) report that embeddings learned from scratch directly on ATIS data (*task-specific embeddings*) are better than

SENNA. However, task-specific embeddings are composed not only by words but also by named entities (*NE*) and syntactic features[1]. NE improves performance significantly while part-of-speech only adds small benefits. Ravuri and Stolcke (2015) emphasizes the importance of *character representation* to handle OOV issues.

For the *encoder layer*, various RNN architectures have been applied to SF and IC (Mesnil et al., 2013, 2015; Liu and Lane, 2015). Mesnil et al. (2013) compare the Elman (Elman, 1990) and Jordan (Jordan, 1997) RNNs. They observe that the performance of the Jordan RNN is marginally better than Elman. They also experiment a *bi-directional* version of Jordan RNN and obtained the best score of 93.89 F1 for SF, performing better than CRF for about +1 absolute F1 improvement. Xu and Sarikaya (2013) use Convolutional Neural Network (CNN) (LeCun et al., 1998) to extract 5-gram features and apply max-pooling to obtain the word representation before passing it to the output layer. Compared with RNN (Yao et al., 2013; Mesnil et al., 2013), CNN gives lower performance for SF on ATIS. Other studies (Yao et al., 2014a; Vu et al., 2016) adapt Long Short-Term Memory Network (LSTM) (Hochreiter and Schmidhuber, 1997) to SF. The LSTM model gives better SF performance compared to CRF, CNN, and RNN. Ravuri and Stolcke (2015) compare the performance of vanilla RNN and LSTM for IC. They find that the vanilla RNN works best for shorter utterances, while LSTM is better for longer utterances.

For the *output layer*, typically a *softmax* function is used for prediction at a particular time step. Yao et al. (2014b) propose a R-CRF model combining the feature learning power of RNN and the *sequence level optimization* of CRF for SF. The RNN + CRF scoring mechanism incorporates the features learned from RNN and the transition scores of the slot slot labels. R-CRF outperforms CRF and vanilla RNN on ATIS and on the Bing query understanding dataset. Table 3.5 summarizes the performance of independent models on SF and IC.

Takeaways on independent SF and IC models:

- Performance of RNN encoders (*unidirectional*) are Jordan $\leq$ Elman $<$ LSTM. Bi-directional encoding is additive to the performance of each encoder.
- Incorporating more context information is better for SF performance. Using global context information, such as sentence level representation, and attention mech-

[1]Gold named entity and syntactic information

	Input	**Model (Enc/Dec)**	**Output**	**Slot (F1) (F1)**	**Intent(Err) (Err)**
Xu and Sarikaya (2013)	lexical	CNN	softmax	94.35	6.65
Yao et al. (2013)	lexical	Elman RNN	softmax	94.11	-
Yao et al. (2013)	lexical+NE	Elman RNN	softmax	96.60 -	
Yao et al. (2014a)	lexical	LSTM	softmax	94.85	-
Yao et al. (2014b)	lexical+NE	Elman RNN	CRF	96.65	-
Mesnil et al. (2015)	lexical	Hybrid Elman + Jordan RNN	softmax	95.06	-
Liu and Lane (2015)	lexical	Elman RNN with label sampling	softmax	94.89	-
Vu et al. (2016)	lexical	bi-directional RNN	softmax	94.92	-
Liu and Lane (2016a)	lexical	bi-directional RNN +attention	softmax	95.75	2.35
Kurata et al. (2016b)	lexical	Encoder-Decoder LSTM	softmax	95.40	-

Table 3.5: Comparison of independent SF and IC models and their performance on ATIS.

anisms (Kurata et al., 2016b; Liu and Lane, 2016a) boosts performance of bi-directional encoder even further.

- When adding external features is possible, semantic features such as NE are more beneficial than syntactic features for SF. When NE is used, it can boost the model performance for SF significantly.

- The slot filling task is related to Named Entity Recognition (NER) (Grishman and Sundheim, 1996) task as slot values can be a named entity such as airline name, city name etc. If the slot filling task is modeled as a sequence tagging problem, basically recent neural models proposed for NER can be used for slot filling and vice versa. To know more about the recent development of neural NER models, one can consult the survey from Yadav and Bethard (2018).

- The main disadvantage of independent models is that they do not exploit the interaction between intent and slots and may introduce error propagation when they are used in a pipeline.

3.5.2 Joint Models

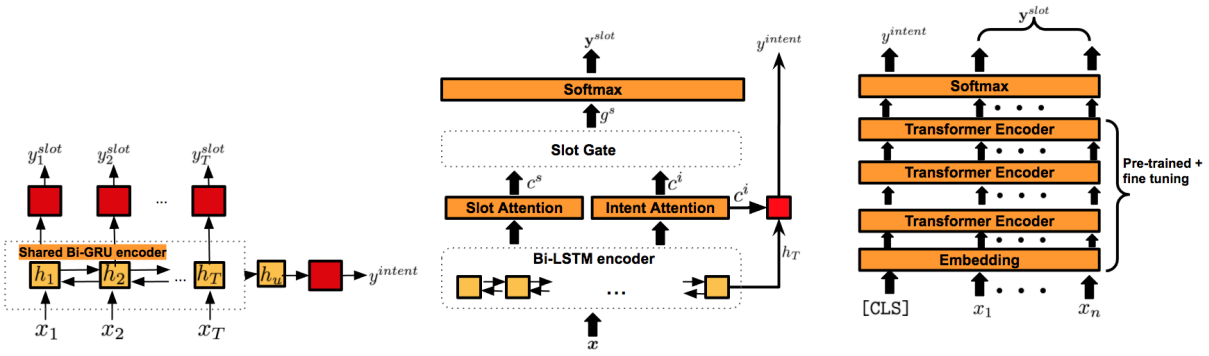


Figure 3.1: *Left:* Shared Bi-GRU encoder (Zhang and Wang, 2016). *Middle:* Slot-Gate Mechanism (Goo et al., 2018). *Right:* BERT Based (Chen et al., 2019).

In Section 3.5.1 we reported approaches that treat SF and IC *independently*. However, as the two tasks always appear together in an utterance and they share information, it is intuitive to think that they can benefit each other. For instance, if the word "*The Beatles*" is recognized as the slot ARTIST, then it is more likely that the intent of the utterance is PLAYSONG rather than BOOKFLIGHT. On the other hand, recognizing that the intent is PLAYSONG would help to recognize "*Hey Jude*" as the slot ARTIST rather than MOVIENAME.

Recent approaches model the relationship between SF and IC *simultaneously* in a *joint model*. These approaches promote *two-way* information sharing between the two tasks instead of a one-way (*pipeline*). We describe several alternatives to exploit the relation between SF and IC: through *parameter and state sharing* and *gate mechanism*.

Parameter and State Sharing. A pioneering work in joint modeling is Xu and Sarikaya (2013), which performs parameter sharing and captures the relation between SF and IC through Tri-CRF (Jeong and Lee, 2008). The model uses CNN as a *shared* encoder for both tasks and the produced hidden states are utilized for SF and IC. In addition to features learned from the NN and from the slot label transition, Tri-CRF incorporates an additional factor g to learn the correlation between the slot label assigned to each word and the intent assigned to the utterance, which explicitly captures the dependency between the two tasks. A similar approach (Guo et al., 2014), shares the node representation produced by Recursive Neural Network (RecNN) which operates on the syntactic tree of

the utterance. The node's representation is *shared* among SF and IC. Zhang and Wang (2016) use a *shared* bi-GRU encoder and a *joint loss function* between SF and IC (Figure 3.1 *Left*), in which the loss function has weights associated with each tasks.

Liu and Lane (2016a) use a neural sequence to sequence (encoder-decoder) model with attention mechanism commonly used for neural machine translation. The *shared* encoder is a bi-directional LSTM, and the last hidden state of the encoder is then used by the decoder to generate a sequence of slot labels, while for IC there is a separate decoder. The attention mechanism is used to learn alignments between slot labels in the decoder and words in the encoder. Hakkani-Tür et al. (2016) also adopt parameter sharing similar to Zhang and Wang (2016), but instead of using GRU they use a shared LSTM and perform predictions for slots, intent, and also domain.

In a recent approach by Wang et al. (2018b) they propose a bi-model based structure to learn the *cross-impact* between SF and IC. They argue that a single model for two tasks can hurt performance, and, instead of sharing parameters, they use two-task networks to learn the cross-impact between the two tasks and only share the hidden state of the other task. In the model, every hidden state h_t^1 in the first network is combined with the hidden state of the second network h_t^2, and vice versa. Training is also done asynchronously, as each model has a separate loss function. Qin et al. (2019) use a self-attentive shared encoder to produce better context-aware representations, then apply IC at the *token level* and use this information to guide the SF task. They argue that previous work based on *single utterance-level* intent prediction is more prone to error propagation. If some token-level intent is incorrectly predicted, the other correct token-level prediction can still be useful for corresponding SF. For the final IC prediction, they use a voting mechanism to take into account the IC prediction on each token.

Chen et al. (2019) use a Transformer (Vaswani et al., 2017) model for joint SF and IC by fine-tuning a pre-trained BERT (Devlin et al., 2019) model (Figure 3.1 *Right*). The input is passed through several layers of transformer encoders and the hidden state outputs are used to compute slot and intent labels. The hidden state $h^{\texttt{CLS}}$ is used for IC[2] while the rest of the hidden states at each time step h_i serve SF.

Slot-Intent Gate Mechanism. In addition to parameter and state sharing, a separate network with a *slot gating mechanism* was introduced by Goo et al. (2018) to model the

[2] `[CLS]` is a special token in BERT input format that often used as the sentence representation.

interaction between SF and IC more explicitly (Figure 3.1 *Middle*). In the encoder, a *slot context vector* for each time step, $\boldsymbol{c}_i^S$, and a global intent context vector $\boldsymbol{c}^I$ are computed using an attention mechanism (Bahdanau et al., 2015). The slot-gate $\boldsymbol{g}^s$ is computed as a function of $\boldsymbol{c}_i^S$ and $\boldsymbol{c}^I$, $\boldsymbol{g}^s = \sum v \cdot tanh(\boldsymbol{c}_i^S + \boldsymbol{W} \cdot \boldsymbol{c}^I)$. Then, $\boldsymbol{g}^s$ is used as a weight between $\boldsymbol{h}_i$ and $\boldsymbol{c}_i^S$ to compute y_i^{slot} as follows: $y_i^{slot} = \texttt{softmax}(\boldsymbol{W}(\boldsymbol{h}_i + \boldsymbol{g}^s \cdot \boldsymbol{c}_i^S))$. Larger $\boldsymbol{g}^s$ indicates a stronger correlation between c_i^S and c^I.

E et al. (2019) propose a bi-directional model, SF-ID (SF-Intent Detection) network, sharing ideas with Goo et al. (2018), with two key differences. First, in addition to the slot-gated mechanism, they add an intent-gated mechanism as well. Second, they use an iterative mechanism between the SF and ID network, meaning that the gate vector from SF is injected into the ID network and vice versa. This mechanism is repeated for an arbitrary number of iteration. Compared to (Goo et al., 2018), the SF-ID network performs better both in SF and IC on ATIS and SNIPS. The work from Li et al. (2018) is also similar to Goo et al. (2018) with two differences. First, they use a self-attention mechanism (Vaswani et al., 2017) to compute c_i^S. Secondly, they use a separate network to compute gate vector g^s, but the input of this network is the concatenation of c_i^S and the intent embedding v, and g^s is defined as $g^s = \texttt{tanh}(\boldsymbol{W}^g[c_{slot}^i, v^{intent}] + b^s)$. After that, h_i is combined with g^s through element-wise multiplication to compute y_i^s as follows: $y_i^{slot} = \texttt{softmax}(\boldsymbol{W}^s(h_i \odot g^s) + b^s)$. They report a +0.5% improvement on SF over Liu and Lane (2016a). A recent work by Zhang et al. (2019), further improves the performance of the BERT based model by adding a gate mechanism (Li et al., 2018) to the BERT model. Table 3.6 compares the performance of the joint models.

Takeaways on joint SF and IC models:

- The overall performance of joint models for SF and IC (Table 3.5) is competitive with independent models (Table 3.6). The advantage of joint models is that they have relatively fewer parameters than independent models, as both tasks are trained on a single model.

- When computational power is not an issue, fine-tuning a pre-trained model such as BERT is the way to go for maximum SF and IC performance. Hybrid methods combining parameter and state sharing + intent gating yield the best performance (Zhang et al., 2019).

Method	Model	ATIS		SNIPS	
		Slot F1	Intent Acc/Err	Slot F1	Intent Acc/Err
Parameter & State Sharing					
Xu and Sarikaya (2013)	CNN + Tri-CRF	95.42	-/5.91	-	-
Guo et al. (2014)	Recursive NN	93.96	95.40	-	-
Zhang and Wang (2016)	Joint Multi-Task,Bi-GRU	95.49	98.10	-	-
Liu and Lane (2016a)	Seq2Seq + Attention	94.20	91.10	87.80	96.70
Hakkani-Tür et al. (2016)	Bi-LSTM	94.30	92.60	87.30	96.90
Qin et al. (2019)	Token-Level IC + Self-Attention	95.90	96.90	94.20	98.00
Chen et al. (2019)	Transformer (BERT)	96.10	97.50	97.00	98.60
State Sharing					
Wang et al. (2018b)	Bi-model, BiLSTM	96.89	98.99	-	-
Slot-Intent Gating					
Goo et al. (2018)	Slot-Gated Full Attention	94.80	93.60	88.80	97.70
Li et al. (2018)	BiLSTM + Self-Attention	96.52	-/1.23	-	-
E et al. (2019)	SF-ID Network	95.75	97.76	91.43	97.43
Hybrid Param Sharing + Gating					
Zhang et al. (2019)	BERT + Intent-Gate	98.75	99.76	98.78	98.96

Table 3.6: Performance comparison of joint models for SF and IC on ATIS and SNIPS-NLU.

- For the non-BERT-based model, using state sharing (Wang et al., 2018b) is the best on ATIS. However, the disadvantage is that it is actually a bi-model and not a single model.

- Similar to independent models, contextual information is crucial to performance. Adding a self-attention mechanism (Qin et al., 2019; Li et al., 2018) to either parameter and state sharing or to slot-intent gating can boost performance even further.

- When sufficiently large in-domain training data is available, the SF and IC performance in ATIS and SNIPS is already saturated. Therefore, further research on this classic leaderboard chase is not worth it. We discuss more about that in Section 6.

- Most of the work in joint models and also independent models (Section §3.5.1) reports F1 scores for slot filling performance. However, these scores do not reveal in which specific cases these models behave differently, contributing to overall performance. We leave further analysis on model performance as a potential future work.

3.6 State of The Art Low-Resource NLU Methods

So far, the models that we consider in Section §3.5.1 and Section §3.5.2 are designed to be trained on a *single domain* (e.g. banking, restaurant reservation, flight booking) and require relatively *large labeled data* to perform well. However, as we have described in §1.2, in practice, new intents and slots are regularly added to a system to support new tasks and *domains* also in *languages* other than English, requiring data and time intensive processes for data collection and annotation. Hence, methods to train models for new domains (*domain scaling*) and languages (*cross-lingual*) with limited or without labeled data are needed.

Generally speaking, there are two common methods to address domain scaling and cross-lingual NLU, namely *transfer learning* and *data augmentation*. In contrast to supervised learning, where we learn a model from scratch, transfer learning (Pan and Yang, 2010; Ruder, 2019) aims to leverage the knowledge learned from related settings (task, domain, and language) to learn a model for a target setting. While for data augmentation, the aim is to automatically extend the original training data on the target setting by performing data transformation. The next subsections discuss the transfer learning and data augmentation approaches applied for domain scaling (§3.6.1) and cross-lingual NLU (§3.6.2) in ToD.

3.6.1 Scaling to New Domains

Setup. The setup of the problem assume that there are K source domains $\mathcal{D}_S^1, \mathcal{D}_S^2, \ldots, \mathcal{D}_S^K$ and a target domain $\mathcal{D}_T^{K+1}$, and there is an abundance of data in $\mathcal{D}_\mathcal{S}$ and limited data in $\mathcal{D}_\mathcal{T}$. For Transfer Learning (TF) approaches, instead of training a target model $\mathcal{M}_T$ for $\mathcal{D}_T$ from scratch, TF aims to *adapt* the learned model $\mathcal{M}_S$ from $\mathcal{D}_S$ to produce a model

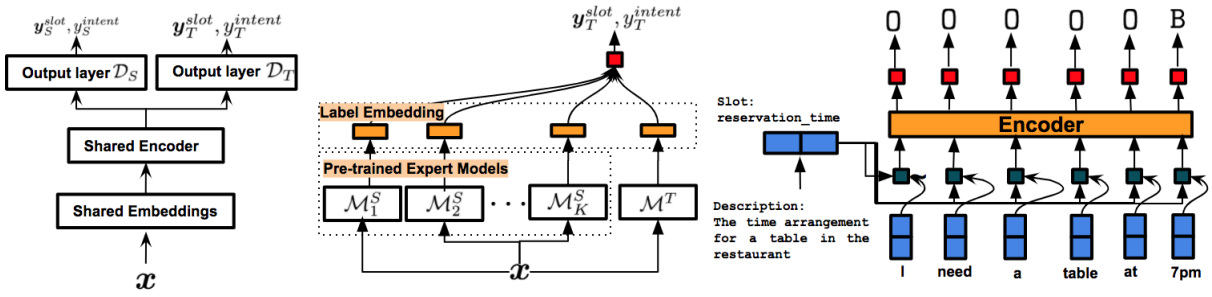


Figure 3.2: *Left:* Data-driven approach (Jaech et al., 2016; Hakkani-Tür et al., 2016). *Middle:* Model-Driven Approach with expert models (Kim et al., 2017). *Right:* Zero-shot model (Bapna et al., 2017).

$\mathcal{M}_T$ trained on $\mathcal{D}_T$. TF is typically applied with various parameter sharing and training mechanisms. As for data augmentation (DA), typically, particular data transformation is applied to $\mathcal{D}_\mathcal{T}$ to produce $\mathcal{D}_\mathcal{T}^{'}$, then the model $\mathcal{M}_T$ is trained on the union of the original dataset and the new synthetic data, $\mathcal{D}_\mathcal{T} \cup \mathcal{D}_\mathcal{T}^{'}$.

For SF and IC two transfer learning based approaches are proposed, namely *data-driven* and *model-driven*. As for data-driven techniques, typically, we combine data from $\mathcal{D}_S$ and $\mathcal{D}_T$ and train the model in a way that allows knowledge sharing between $\mathcal{M}_S$ and $\mathcal{M}_T$.

Data-driven via Multi-Task Learning. Some studies (Jaech et al., 2016; Hakkani-Tür et al., 2016) apply knowledge sharing using *multi-task learning* (MTL) (Caruana, 1997). In MTL, the parameters of the model are partitioned into parts that are *task-specific* and parameters that are shared across tasks (Figure 3.2 *Left*). The MTL model is typically trained in an alternating fashion on $\mathcal{D}_S$ and $\mathcal{D}_T$. Results have shown that MTL is particularly effective relative to single-task learning (STL) when the data in $\mathcal{D}_\mathcal{T}$ is scarce, and the benefits over STL diminish as more data is available. One aspect that is believed to be important is the characteristic of auxiliary tasks that are used in MTL. It is expected that the auxiliary tasks should be similar to the target task for MTL to be helpful for the target task. There are other approaches from Hakkani-Tür et al. (2016) that use MTL, but they focus on improving performance on multi-domain settings where sizeable data for each domain is already available. Most approaches (Jaech et al., 2016; Hakkani-Tür et al., 2016) use the data from $\mathcal{D}_S$ as auxiliary tasks, i.e., conversational text labeled NLU data.

Data-driven via Pre-train & Fine Tuning. Another technique that is typically used in data-driven approaches is based on *pre-train* and *fine-tune* mechanisms. In contrast to MTL, the general idea of pre-train and fine-tuning is first to train a model $\mathcal{M}$ on $\mathcal{D}_S$ then uses the learned weight to initialize $\mathcal{M}_T$ and continue the training on $\mathcal{D}_T$. Goyal et al. (2018) train a joint model of SF and IC, $\mathcal{M}_S$, on large $\mathcal{D}_\mathcal{S}$, then fine-tune $\mathcal{M}_S$ by replacing the output layer corresponding with the label space from $\mathcal{D}_T$ and train the model further on $\mathcal{D}_T$. Siddhant et al. (2019) also uses fine-tuning mechanism, but the main difference with Goyal et al. (2018) is they leverage large *unlabeled data* to learn contextual embedding, ELMo (Peters et al., 2018), before fine-tuning on $\mathcal{D}_T$.

As we need to train from scratch the whole model when adding a new domain, data-driven approaches, especially MTL-based, need increasing training time as the number of domains grows. The alternative strategy, the model-driven approach, alleviates the problem by enabling model *reusability*. Although different domains have different slot schemas, slots such as *date*, *time* and *location* can be shared.

Model-driven via Expert-based. In model-driven adaptation, "expert" models (Figure 3.2 *Middle*) are first trained on these reusable slots (Kim et al., 2017; Jha et al., 2018) and the outputs of the expert models are used to guide the training of $\mathcal{M}_T$ for a new target domain. This way, the training time of $\mathcal{M}_T$ is faster, as it is proportional to the $\mathcal{D}_T$ data size, instead of the larger data size of the whole $\mathcal{D}_S$ and $\mathcal{D}_T$. In this model-driven setting, Kim et al. (2017) do not treat each expert model on each $\mathcal{D}_S$ equally; instead, they use an attention mechanism to learn a weighted combination from the feedback of the expert models. Jha et al. (2018) use a similar model as Kim et al. (2017); however, they do not use an attention mechanism. For training the expert models, instead of using all available $\mathcal{D}_S$, they build a repository consisting of common slots, such as *date*, *time*, *location* slots. The assumption is that these slots are potentially reusable in many target domains. Upon training $\mathcal{M}_S$ on this reusable repository, the output of $\mathcal{M}_S$ is directly used to guide the training of $\mathcal{M}_T$.

Zero-shot. While data-driven and model-driven approaches can share knowledge learned on different domains, such models are still trained on a pre-defined set of labels, and can not handle *unseen* labels, i.e., not mapped to the existing schema. For example, a model trained to recognize a DESTINATION slot can not be used directly to recognize the slot ARRIVAL_LOCATION for a new domain, although both slots are semantically similar. For

this reason, researchers have recently been working on *zero-shot* models, trained on *label representations* that leverage natural language *descriptions* of the slots (Bapna et al., 2017; Lee and Jha, 2019). Assuming that accurate slot descriptions are provided, slots with *different* names, although semantically similar, would have a similar description as well. Thus, having trained a model for the DESTINATION slot with its descriptions, it is now possible to recognize the slot ARRIVAL_LOCATION without training on it, but only supplying the corresponding slot description.

In addition to slot description, other zero-shot approaches explore the use of slot value examples (Shah et al., 2019; Guerini et al., 2018). Shah et al. (2019) show that a combination of a small number of slot values examples with a slot description performs better than (Bapna et al., 2017; Lee and Jha, 2019) on the SNIPS dataset. Zero-shot models are typically trained on a per-slot basis (Figure 3.2 *Right*), meaning that if we have N slots, then the model will output N predictions. Therefore, a merging mechanism is needed in case there are prediction overlaps. In order to alleviate the problem of having multiple predictions, Liu et al. (2020b) propose a *coarse-to-fine* approach, in which the model learns the slot entity pattern (coarsely) to identify a particular token is an entity or not. After that, the model performs a single prediction of the slot type (fine) based on the similarity between the feature representation and the slot description.

Data Augmentation. As mentioned before, Transfer Learning (TF) primarily focuses on the mechanism on transferring learned representation from source settings to target settings, while data augmentation (DA), on the other hand, aims to generate *meaning preserving* additional labeled data $\mathcal{D}'_{\mathcal{T}}$. DA has been commonly used, for example, in Computer Vision research (Krizhevsky et al., 2012; Summers and Dinneen, 2019) where operations such as image flipping, cropping, color jittering have been a standard technique. However, in the context of NLP, DA is less trivial as the input space is discrete, and it is still challenging to produce meaning-preserving labeled data. Despite the challenges and lack of standard augmentation, there has been an increasing interest in applying data augmentation in NLP. Different techniques have been proposed from relatively simple rule-based methods (*heuristic*) such as word substitution, deletion, addition (Wang and Yang, 2015; Wei and Zou, 2019b) until model-based approaches that train a neural model or pre-trained model to generate synthetic text (Kobayashi, 2018; Kumar et al., 2020; Anaby-Tavor et al., 2020). DA has shown potential in low-resource NLP tasks such as

text classification (Wei and Zou, 2019b), parsing (Sahin and Steedman, 2018; Vania et al., 2019a), and machine translation (Sennrich et al., 2016; Fadaee et al., 2017).

In the context of NLU in ToD, a number of DA approaches have been proposed to generate synthetic labeled utterances. Kurata et al. (2016a) trains an LSTM encoder-decoder model on the *same* utterance as input and output sequences. In the augmentation process, the model encodes the original utterance, and they perform perturbation (additive, multiplicative perturbation) on the hidden state to generate different tokens and slot labels in utterance. The work from Hou et al. (2018b) trains a sequence to sequence (Seq2Seq) model (Sutskever et al., 2014) on a pair of utterances that belongs to the same semantic frame (intent) cluster. Similar to Kurata et al. (2016a), they use the last hidden state of the Seq2Seq encoder to generate the synthetic utterance. They also incorporate diversity measures to generate more varied utterances. Zhao et al. (2019) also uses Seq2Seq model as Hou et al. (2018b), but they incorporate intermediate atomic templates that provide a mapping between the semantic representation and the raw natural language utterance. The work from Yoo et al. (2019) follows the same idea, but they use a Variational Auto Encoder (VAE) (Kingma and Ba, 2015) to generate additional labeled utterances.

With the advent of pre-trained language models such as BERT (Devlin et al., 2019), BART (Lewis et al., 2020), and GPT-2 (Radford, 2018), pretrained on language modeling tasks on massive amount of data, these off-the-shelf models have been used to obtain better *contextual augmentation*. In contrast to most prior approaches in which the augmented words are limited on the words that appear in $\mathcal{D}_T$, these pre-trained models can produce words outside $\mathcal{D}_T$. Anaby-Tavor et al. (2020) fine-tunes GPT-2 on $\mathcal{D}_T$ with the following format: y_1 `SEP` x_1 `EOS` y_2 `SEP` x_2 `EOS` ...y_n `SEP` x_n `EOS`, where y_i is the intent label of the utterance x_i, `SEP` indicates the separator between the intent label and the utterance, and `EOS` marks the end of the utterance. After fine-tuning, to generate a synthetic utterance, we supply an intent label and a separator "y `SEP`" then let the model generate the utterance token by token until `EOS`. The approach from Peng et al. (2020) also uses GPT-2 as Anaby-Tavor et al. (2020) but with different pre-training and fine-tuning strategies. They continue pre-training GPT-2 on unlabeled 400K dialogue corpus and then fine tuning on two scenarios: fine-tune on a dataset where the ontology (intent and slot values annotation) is available or on a dataset where domain-specific unlabeled dataset is available.

Kumar et al. (2020) compares several Transformer models: BERT, BART, and GPT-2[3], fine-tune each model on $\mathcal{D}_T$ conditioned on the intent label and generate the synthetic utterance. They found that BART works the best for intent classification on the SNIPS dataset.

Summary. In this subsection, we have seen that there have been a wide variety of approaches to domain scaling. While all of the approaches share the same motivation to overcome limited labeled data on the target setting, each of the approaches has different requirements and assumptions regarding available auxiliary data or models that can be used as a source of transfer or augmentation. Table 3.7 compares the requirements of auxiliary data or pre-trained model and the mechanism for transfer learning or augmentation from each approach.

Takeaways on scaling to new domains with transfer learning.

- Both data driven methods, MTL and pre-train fine tuning, improve performance when data in $\mathcal{D}_T$ is limited. Both are also flexible, as virtually many tasks from different domains can be plugged into these methods. As the number of domains grow, pre-train and fine tuning is more desirable than MTL. However, fine tuning is more prone to the *forgetting* problem (He et al., 2019) compared to MTL.
- When the number of domains, K, is massive, the pre-train fine tuning approach and model driven approaches, such as expert based adaptation, are preferable with respect of training time.
- When there exists K existing domains and no annotation is available in $\mathcal{D}_T$, the choice is zero-shot approaches with the expense of providing meta-information such as slot and intent descriptions.

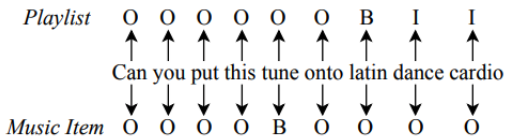


Figure 3.3: Example of zero-shot predictions for slot filling in an utterance. Prediction is performed on a per slot basis. Figure is taken from Liu et al. (2020b).

- Typically zero-shot models perform prediction on a *per-slot* basis (Figure 3.3). Given

[3]Similar to Anaby-Tavor et al. (2020) but instead of using "y SEP" prompt for generation, they add more context words i.e., "y SEP $w_1 w_2 w_3$"

	Method	Auxiliary Requirements
Transfer Learning		
Jaech et al. (2016)	Multi-Task Learning	Labeled conversational slot filling $\mathcal{D}_S$
Goyal et al. (2018)	Pre-train & Fine-Tuning	Labeled conversational slot filling $\mathcal{D}_S$
Siddhant et al. (2019)	Pre-train & Fine-Tuning	Unlabeled conversational $\mathcal{D}_S$ Labeled conversational slot filling $\mathcal{D}_S$
Kim et al. (2017)	Expert model	Labeled conversational slot filling $\mathcal{D}_S$ Pre-trained models on $\mathcal{D}_S$
Jha et al. (2018)	Expert model	Coarse-grained labeled conversational slot filling $\mathcal{D}_S$ Pre-trained models on $\mathcal{D}_S$
Bapna et al. (2017)	Train on $\mathcal{D}_S$ only (zero-shot)	Labeled conversational slot filling $\mathcal{D}_S$ Natural language description of slot names
Lee and Jha (2019)	Train on $\mathcal{D}_S$ only (zero-shot)	Labeled conversational slot filling $\mathcal{D}_S$ Natural language description of slot names
Shah et al. (2019)	Train on $\mathcal{D}_S$ only (zero-shot)	Labeled conversational slot filling $\mathcal{D}_S$, Natural language description of slot names Slot value examples
Guerini et al. (2018)	Train on $\mathcal{D}_S$ only (zero-shot)	Slot value examples
Data Augmentation		
Kurata et al. (2016b)	Model based DA	—
Hou et al. (2018b)	Rule, Model based DA	—
Zhao et al. (2019)	Model based DA	Intent and slot Value Template
Yoo et al. (2019)	Model based DA	—
Peng et al. (2020)	Model based DA	Unlabeled conversational $\mathcal{D}_S$ Pre-trained GPT-2
Anaby-Tavor et al. (2020)	Model based DA	Pre-trained GPT-2
Kumar et al. (2020)	Model based DA	Pre-trained BART, BERT, GPT-2

Table 3.7: Comparison of transfer learning and data augmentation approaches on domain scaling focusing on the methods and their auxiliary requirements. For more comparison in terms of the neural models, evaluated tasks, and type of augmentation, see Table A.1 .

an utterance and K possible slots, prediction is performed K times. Consequently, a token in the utterance can be classified into more than one slot. For example, in Figure 3.3, the word "tune" can be predicted as *Playlist* and *Music Item* slots which add more difficulties for the final prediction (Liu et al., 2020b). In addition to that, per-slot

prediction can also be computationally inefficient, especially when there are many slots and utterances to process.

Takeaways on scaling to new domains with data augmentation.

- While numerous methods have been proposed, they share similar goals that are to generate *meaning-preserving* and *diverse* synthetic utterances.
- There are different *granularities* in which the augmentation is applied: on the word-level or sentence-level.
- Recent approaches typically use pre-trained language models such as GPT-2, BERT to perform contextual augmentation. Being pre-trained on massive data, pre-trained LMs may produce words that do not appear in $\mathcal{D}_T$. Moreover, as the pre-training task is a language model task, pre-trained LMs can produce words that fit in the context of the original utterance. However, it is still relatively challenging to generate label-preserving utterance or slot values.
- Another drawback of data augmentation is that it increases the training time of the NLU model as we add synthetic utterances. In addition to that, model-based approaches consist of several stages, such as ranking or filtering utterances to improve diversity and preserve original meaning, which can contribute to more training time.

3.6.2 Cross-Lingual NLU

As we have described in Section 1.3, most NLU approaches are focused on major languages, e.g., English, and it is still a challenge in NLU, or even other dialogue sub-tasks is to achieve *multilingual* ToD systems that support many languages. As the supervised learning model obtains the best performance in NLU, the bottleneck is to obtain a sufficiently large labeled dataset for many languages. Collecting sizeable labeled data for many languages is infeasible. This data scarcity drives most NLU research towards *cross-lingual* transfer (zero-shot or few shots) approaches. In zero-shot cross-lingual transfer, it is typically assumed that only a high-resource language, e.g., English labeled data is available for training the model and transfer directly to other languages. This section reviews existing approaches on cross-lingual NLU.

Transfer Learning via Cross-Lingual Word Embedding. Upadhyay et al. (2018) uses the monolingual embedding from `fastText`(Bojanowski et al., 2017) and project em-

beddings from two different languages into a shared semantic space to produce bilingual embeddings through a linear transformation (Smith et al., 2017). After that, they train an NLU model using the bilingual embeddings on the English labeled data and perform the zero-shot evaluation. Liu et al. (2019c) refines a cross-lingual word embeddings from Joulin et al. (2018) using a self-learning framework (Artetxe et al., 2017) by providing a small number of English in-domain lexicon. In their subsequent work Liu et al. (2020c) uses their refined cross-lingual word embedding and add *label regularization* method to improve the cross-lingual alignment. They do this by taking into account the slot label sequence to produce better utterance representation. The intuition is that if two utterances have a similar slot label sequence, then both utterances have a similar meaning as well.

Transfer Learning via Massive Multilingual Transformer (MMT) Model. Several works have investigated the multilingual aspect of a pre-trained multilingual pre-trained model such as multilingual BERT (mBERT) (Devlin et al., 2019) and XLM-R (Lample and Conneau, 2019). These models are trained on a concatenated unlabeled data from multiple languages. For example, mBERT is trained on Wikipedia data from 104 languages. Despite *no cross-lingual supervision* and rely on a language modeling objective, multilingual pre-trained models have shown potential for zero-shot cross-lingual transfers on NER, POS tagging, dependency parsing, and Natural Language Inference tasks (Wu and Dredze, 2019; Pires et al., 2019). Motivated by these developments, recently, the default baseline approach for zero-shot cross-lingual has been fine-tuning a multilingual pre-trained model such as mBERT (Devlin et al., 2019), XLM-R (Lample and Conneau, 2019) on the English labeled dataset. Despite its potential for zero-shot transfer, it is known that MMT is less effective to distant target languages —from English— and for languages that have less amount of data for pre-training (Lauscher et al., 2020).

Improving Contextual Cross-Lingual Representation. As MMTs, such as mBERT, XLM-R, do not include cross-lingual alignment between languages in the pre-training stages, recent approaches have been focusing on injecting cross-lingual supervision to improve performance on the cross-lingual transfer tasks. Most methods are characterized by different resource requirements such as the availability of parallel data or machine translation model and in which stages the cross-lingual supervision occurs, such as during fine-tuning or pre-training or in between the two stages. Schuster et al. (2019a) uses a Neural Machine Translation (NMT) system to produce utterances on the target language.

Then annotation projection is performed to project slot labels from the original utterance to the target utterance. For the annotation projection, they use the attention weights of their NMT system. Xu et al. (2020) also uses an NMT system to generate utterances on the target language. However, instead of using attention weights of the NMT, they propose an end-to-end model that learns the slot alignments through an attention module and also reconstructs the original utterance. Kulshreshtha et al. (2020) compares several cross-lingual alignment methods with different cross-lingual sources such as bilingual dictionary and parallel corpus. They report that fine-tuning mBERT on a word alignment task from a parallel corpus significantly improves zero-shot transfer on several downstream tasks, including slot filling. van der Goot et al. (2021) use multi-task learning to compare the usefulness of several auxiliary tasks namely Mask Language Modeling (MLM), Neural Machine Translation (NMT), and Universal Dependency (UD) parsing for zero-shot NLU. They found that multi-task with Mask Language Modeling (MLM) is the most robust for slot filling but it does not help intent classification.

Data Augmentation. Another line of work on improving cross-lingual representation involves data augmentation approaches to generate *code switched* data in which two or more languages are used in the same utterance. Qin et al. (2020a) improves the multilingual representation of mBERT by replacing words in the English utterance with a word from one of the target languages during fine-tuning. They define a threshold ratio that controls how many sentences and words are replaced during the fine-tuning stage. Liu et al. (2020a) also performs selective code-switching (CS) by using attention weights to choose which words to be replaced. Both approaches (Qin et al., 2020a; Liu et al., 2020a) use a bi-lingual dictionary to look up the substitute words.

Summary. In this subsection, we have surveyed the literature related to cross-lingual NLU, focusing on zero-shot cross-lingual. Similar to the domain scaling problem, zero-shot cross-lingual NLU also adopts transfer learning and data augmentation techniques. Overall, most of the approaches can be characterized by the method to improve the cross-lingual representation and its auxiliary requirements. Table 3.8 summarizes the comparison across approaches.

Takeaways on zero-shot cross-lingual NLU.

- In terms of SF and IC performance, methods that give the best performance typically involve the translate and train method combined with slot labels projection (Schuster

	Method	**Auxiliary Requirements**
Upadhyay et al. (2018)	Embedding alignment	Monolingual word embedding
Liu et al. (2019c)	Embedding alignment	Cross-lingual word embedding Slot value examples
Liu et al. (2020a)	Code switching via attention	Bilingual dictionary
Liu et al. (2020c)	Label regularized alignment	Cross-lingual Word Embedding
Schuster et al. (2019a)	Translate + Train Annotation Projection	MT Model, pre-trained MMT
Xu et al. (2020)	Translate + Train E2E Projection	MT Model, pre-trained MMT
Kulshreshtha et al. (2020)	Embedding alignment	Pre-trained MMT Parallel corpus Bilingual dictionary Word aligner
Qin et al. (2020a)	Code switching	Bi-lingual dictionary

Table 3.8: Comparison of zero-shot cross-lingual NLU approaches focusing on the method and its auxliary requirements.

et al., 2019a; Xu et al., 2020). These methods require the availability of MT models for each target language.

- The current de-facto approach for zero-shot cross-lingual NLU is to fine-tune MMT such as BERT and XLM-R on the English SF and IC datasets. Given that no cross-lingual supervision is performed during the pre-training of an MMT, many methods have been proposed to improve MMT's cross-lingual representation. These methods include cross-lingual embedding alignment leveraging a bi-lingual dictionary or parallel corpus before MMT fine-tuning and code-switching during the fine-tuning stage.
- In the case of MMT fine-tuning, it could be the case that using English as the source of transfer is not always optimal in zero-shot settings. Providing a small number of examples (*few-shot*) may yield a larger benefit than zero-shot as suggested by Lauscher et al. (2020).

3.7 Conclusion & Context on Contributions

In this chapter we have introduced in more detail both the background and the definition of NLU in the context of ToD which is based on intent classification and slot filling tasks. Several neural NLU models have been proposed, which can be categorized into *independent* and *joint models*. As shown in Table 3.6, the performance of these state-of-the-art models on standard benchmark datasets such as ATIS and SNIPS have been relatively saturated. Evaluation on these standard benchmarks is typically performed by training an individual model, which assumes that sizeable training data is available for each dataset. However, as we have described in §3.6 new domains and languages are regularly added to ToD systems, and adding relatively large labeled data for each new domain and language is expensive. Therefore, methods to scale NLU models to new domains and languages with limited labeled data (*low-resource*) are necessary.

We have described various research efforts to solve low-resource NLU through transfer learning and data augmentation. Approaches can be characterized by their specific *methods*, such as multi-task learning, pre-train & fine tuning, model based augmentation, etc., and by assuming the *availability of auxiliary requirements*, such as the availability of labeled conversational dataset on source domains, pre-trained language models, unlabeled conversational data, machine translation models etc. (Table 3.7 and Table 3.8). The contributions of this thesis, **C1**, **C2**, **C3** from §1.4 are built based on top of existing studies on transfer learning and data augmentation for low-resource NLU.

On the contribution **C1** Leveraging Non-Conversational Text As a Source of Transfer (Chapter 4), we investigate an alternative auxiliary data that can be used for low-resource NLU, specifically slot filling tasks. From Table 3.7 it is evident that labeled conversational dataset in the source domain is typically assumed to be available as auxiliary data. While this assumption is reasonable, it does not cover the cold-start situation in which no labeled conversational dataset is available as auxiliary data. To this end, in our work for **C1**, we investigate the effectiveness of using *non-conversational text* which is annotated with semantically similar tasks with slot filling as auxiliary data. For the contribution **C2** Generating Additional Labeled Data via Lightweight Augmentation (Chapter 5), in contrast to existing data augmentation methods in Table 3.7 that are model-based, we propose a set of simple yet effective augmentation operations. We consider our augmentation operations as *lightweight augmentation* as they do not require training a separate

deep learning model to generate synthetic data. The contribution **C3** Continued Pre-training for Zero-shot Cross Lingual NLU (Chapter 6) is related to zero-shot cross-lingual NLU. The default approach for zero-shot cross-lingual NLU involves fine-tuning a massive multilingual transformer (MMT) pre-trained model (Table 3.8) on the English labeled dataset. We investigate the benefit and analyze important factors of continued domain adaptive pre-training on intermediate unlabeled data before the fine-tuning stage.

Chapter 4

Leveraging Non-Conversational Text

As we have seen in §3.6.1, transfer learning is one of the standard methods used for domain scaling with limited labeled data on task-oriented NLU. Approaches in transfer learning can be characterized by the adaptation technique, e.g., multi-task learning, pre-train fine-tuning, and *auxiliary requirements* that are assumed to be available, e.g., labeled conversational slot filling data (Table 3.7).

This chapter discusses our investigation on alternative auxiliary data that is used on a transfer learning method for low-resource slot filling[1]. We examine a setting where no conversational slot filling data is available as the source of transfer (*cold start*). To this end, we leverage *non-conversational text* (contribution **C1**), which is annotated with similar tasks as slot filling, through multi-task learning (MTL) (Louvan and Magnini, 2019) (§4.1 - §4.4). Furthermore, we investigate whether performing data selection on the auxiliary data before MTL can further boost performance (Louvan and Magnini, 2020a) (§4.5).

4.1 Introduction

Existing works in low-resource slot filling are mostly based on transfer learning (Mou et al., 2016), whose aim is to leverage relatively large resources in a source domain ($\mathcal{D}_S$)

[1]We have related preliminary work in (Louvan and Magnini, 2018a,b) but in this chapter we focus on (Louvan and Magnini, 2019) and (Louvan and Magnini, 2020a)

Sentence	what	is	the	most	expensive	flight	from	boston	to	dallas
ATIS Slot	O	O	O	B-COST_REL	I-COST_REL	O	O	B-FROM_LOC	O	B-TO_LOC
NER	O	O	O	O	O	O	O	B-GPE	O	B-GPE
SemTag	B-QUE	B-ENS	B-DEF	B-TOP	B-IST	B-CON	B-REL	B-GPE	O	B-GPE

Table 4.1: An example of slot filling annotation from the ATIS (Airline Travel Information System) dataset and author-annotated NER and SemTag in IOB format (Ramshaw and Marcus, 1995a). Some ATIS slots correspond to NER or SemTag labels, such as FROM_LOC and TO_LOC with GPE in NER and SemTag. Some slot tags can also be composed of several SemTag labels such as COST_REL which is composed of TOP (*superlative positive*) and IST (*intersective adjective*).

for a source task ($\mathcal{T}_S$), to help a task ($\mathcal{T}_T$) in a target domain ($\mathcal{D}_T$), where less data are available. As described in §3.6.1, depending on how the adaptation is performed, there are two notable approaches: data-driven adaptation (Jaech et al., 2016; Goyal et al., 2018; Kim et al., 2016), and model-driven adaptation (Kim et al., 2017; Jha et al., 2018). Essentially, both approaches produce a model on the target domain performing training on the same task (slot filling, in our case), i.e., assuming $(\mathcal{T}_S = \mathcal{T}_T)$, although from different domains, i.e. $(\mathcal{D}_S \neq \mathcal{D}_T)$. All of these approaches assume that slot filling datasets for the source domain are available, and little effort has been devoted in finding and exploiting cheaper $\mathcal{T}_S$, which is crucial in a situation where a slot filling dataset in $\mathcal{D}_S$ is not ready yet (*cold-start*).

Accordingly, we attempt to leverage non-conversational source tasks ($\mathcal{T}_S \neq \mathcal{T}_T$) i.e., tasks that use widely available non-conversational resources, to help slot filling. These resources are cheaper to obtain compared to domain-specific slot filling datasets, and many of them are annotated with rich linguistic knowledge, which is potentially useful for slot filling (Chen et al., 2016). Among these resources, we mention PropBank (Palmer et al., 2005) and FrameNet (Baker et al., 1998), which consist of annotated documents with verb and frame-based semantic roles, respectively; CoNLL 2003 (Sang and Meulder, 2003) and OntoNotes (Pradhan et al., 2013), which provide named entity information; and Abstract Meaning Representation (AMR) (Banarescu et al., 2013), which provides a graph-based semantic formalism.

In this work, we leverage non-conversational tasks as auxiliary tasks in a multi-task

learning (MTL) (Caruana, 1997) setup. Given appropriate auxiliary tasks, MTL has shown to be particularly effective in which labeled data is scarce and has been applied to various NLP tasks such as parsing (Søgaard and Goldberg, 2016a), POS tagging (Yang et al., 2016), neural machine translation (Luong et al., 2016), and opinion role labeling (Marasovic and Frank, 2018). While there are potentially many non-conversational tasks that we can use as auxiliary tasks, we focus on those that assign semantic class categories to a word, as they are similar in nature to slot filling. In particular, in this work we choose Named Entity Recognition (NER) and the recently introduced Semantic Tagging (SemTag) (Abzianidze and Bos, 2017), motivated by the following rationales:

- Both NER and SemTag are semantically related to slot filling. As illustrated in Table 4.1, slot labels may correspond to either NER or SemTag labels. In addition, SemTag complements NER as its labels subsume NER labels, and thus could be useful to address linguistic phenomena (e.g. comparative expression, intersective adjective) relevant for slot filling and that are beyond named entities.
- Both NER and SemTag can be re-used in many slot filling domains. Labels in both tasks are typically more general (coarse-grained) compared to labels in slot filling.
- The resources for both tasks are cheaper to obtain compared to domain-specific slot filling datasets, as there have been several initiatives in constructing large datasets for NER and SemTag, for example OntoNotes (Pradhan et al., 2013) and Parallel Meaning Bank (PMB) (Abzianidze et al., 2017) respectively. This is beneficial in a *cold-start* situation in which no slot filling dataset is already available in $\mathcal{D}_S$.

Although NER has been already used in slot filling models, most of these approaches (Mesnil et al., 2013, 2015; Zhang and Wang, 2016; Gong et al., 2019; Louvan and Magnini, 2018b) use and incorporate ground truth NER labels or output of NER systems as features to train a slot filling model, our work differs in the method of learning and leveraging such features from disjoint datasets through MTL and evaluating the performance in low-resource settings.

Our contributions are:

- We propose to leverage non-conversational tasks, namely NER and SemTag, to improve low-resource slot filling through MTL; to our knowledge this MTL combination

has not been explored before.

- We show that MTL models with NER and SemTag strongly improve single-task slot filling models on three well known datasets.

While we focus on using NER and SemTag, our study has shed light on the potential use of non-conversational tasks in general to help low-resource slot filling.

4.2 Approach

Slot filling is often modeled as a sequence labeling problem. Given a sequence of words $\mathbf{x} = (x_1, x_2, ..., x_n)$ as input, a model $\mathcal{M}$ predicts the corresponding slot labels $\mathbf{y} = (y_1, y_2, ..., y_n)$ as output.

4.2.1 Base Model

State-of-the-art models on sequence labeling are typically built based on bi-directional LSTM (bi-LSTM), on top of which there is a CRF model (Lample et al., 2016; Ma and Hovy, 2016). The bi-LSTM takes $\mathbf{x}$ as input and each word x_i is represented as an embedding $\mathbf{e}_i = [\mathbf{w}_i; \mathbf{c}_i]$ composed of the concatenation of a word embedding $\mathbf{w}_i$ and character embeddings $\mathbf{c}_i$. The bi-LSTM layer produces the forward output state $\overrightarrow{\mathbf{h}_i}$ and the backward output state $\overleftarrow{\mathbf{h}_i}$. The concatenation of the output states, $\mathbf{h}_i = [\overrightarrow{\mathbf{h}_i}; \overleftarrow{\mathbf{h}_i}]$, is then fed to a feed-forward (FF) layer, followed by a CRF as the final output layer that predicts a slot label y_i by taking into account the mixture of context information captured by the last FF layer and the slot prediction y_{i-1} from the previous word.

4.2.2 Multi-task Learning Models

In the context of MTL, models for $\mathcal{T}_S$, often referred as **auxiliary tasks**, and for $\mathcal{T}_T$, referred as the **target task**, are simultaneously trained (Yang et al., 2017). In order to perform adaptation, the MTL model $\mathcal{M}$ is partitioned into task-specific parts ($\mathcal{M}_{\mathcal{T}_S}$ and $\mathcal{M}_{\mathcal{T}_T}$) and task-shared-parts ($\mathcal{M}_{\mathcal{T}_S \cap \mathcal{T}_T}$). We use two notable MTL architectures:

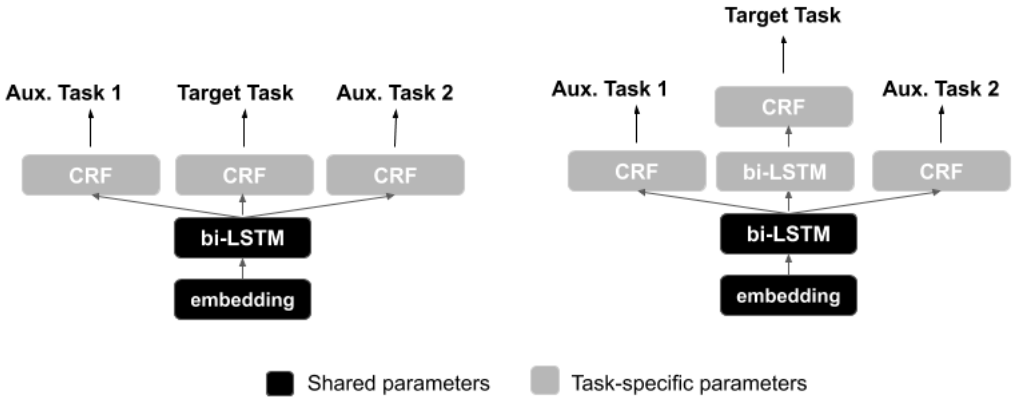


Figure 4.1: Multi-task learning (MTL) models: MTL Fully Shared Network (*Left*) and Hierarchical MTL (*Right*).

- **MTL-Fully Shared Network (MTL-FSN).** This architecture has been explored in NLP including for sequence tagging tasks (Collobert et al., 2011; Søgaard and Goldberg, 2016b; Plank et al., 2016; Changpinyo et al., 2018). The word and character embeddings, and the bi-LSTM layers, are parts of $\mathcal{M}_{\mathcal{T}_S \cap \mathcal{T}_T}$. The hidden state outputs of the bi-LSTM are passed to each of the CRF output layers in $\mathcal{M}_{\mathcal{T}_S}$ and $\mathcal{M}_{\mathcal{T}_T}$. During training a mini-batch of a particular task, the output layers of other tasks are not updated.

- **Hierarchical-MTL (H-MTL).** Inspired by (Søgaard and Goldberg, 2016a; Sanh et al., 2019), we introduce a hierarchy of tasks in $\mathcal{M}$ to create different levels of supervision. Instead of placing the output CRF layers for all tasks after the shared bi-LSTM layer, we add a task-specific bi-LSTM in $\mathcal{M}_{\mathcal{T}_T}$ after the shared bi-LSTM and then attach the output layer. In other words, we supervise $\mathcal{T}_S$, which have coarse-grained labels in the lower level output layer and $\mathcal{T}_T$, which has more fine-grained labels in the higher level output layer.

4.3 Experiments

The main objective of our experiments is to validate the hypothesis that using non-conversational tasks as auxiliary tasks in a MTL setup can help low-resource slot filling. In our MTL configuration, the **target task ($\mathcal{T}_T$)** is slot filling, and the **auxiliary tasks**

($\mathcal{T}_S$) are set to NER or SemTag or both.

Baselines. We compare the two MTL approaches (see §4.2.2) with the following baselines:

- **Single-Task Learning (STL).** The base model is directly trained and tested on $\mathcal{T}_T$, without incorporating any information from $\mathcal{T}_S$. The base model (see §4.2.1) is a bi-LSTM-CRF which is the core of many models for slot filling (Goo et al., 2018; Wang et al., 2018a; Liu and Lane, 2016b) and sequence tagging tasks in general.
- **STL + Feature Based (STL + FB).** The same model as STL but incorporating the outputs of the independently trained NER and SemTag models as an additional feature in the input embeddings. The features are added by concatenating them with the original input embedding.

Datasets. The language of all the datasets that we use is English. We evaluate our approach on three slot filling datasets, namely ATIS (Price, 1990), MIT Restaurant, and Movie (Liu et al., 2013). ATIS is a widely used dataset for spoken language understanding which contains utterances requesting flight related information. While MIT Restaurant and Movie contain utterances requesting information related to restaurants and movies. For NER, we use the newswire section of OntoNotes 5.0 (Pradhan et al., 2012), which is compiled from English Wall St. Journal. For SemTag, we use Parallel Meaning Bank (PMB) (Abzianidze et al., 2017) 2.2.0. The PMB dataset is constructed from twelve different sources, including OPUS News Commentary (Tiedemann, 2012), Tatoeba[2], Sherlock-Holmes stories, Recognizing Textual Entailment (Giampiccolo et al., 2007), and the bible (Christodoulopoulos and Steedman, 2015). Following the previous publication related to SemTag (Abzianidze and Bos, 2017), we train the SemTag model using the silver data[3] and test on gold data. For all datasets, we use the provided train/dev/test splits. Table 4.2 and Table 4.3 shows the overall statistics of each dataset and example sentences and their annotation, respectively. To simulate the low-resource settings, in all experiments we only use 10% randomly sampled training data on $\mathcal{T}_T$.

Training. We do not tune the hyperparameters[4] but follow the suggestions and adapt

[2]https://tatoeba.org/eng/

[3]The silver data consists of the documents tagged with semantic parser from (Bjerva et al., 2016) and some manual corrections

[4]The hyperparameters are listed in Table A.5

Dataset	Task	#train	#dev	#test	#label
ATIS	Slot Filling	4478	500	893	79
MIT Restaurant	Slot Filling	6128	1532	3385	8
MIT Movie	Slot Filling	7820	1955	2443	12
OntoNotes 5.0	NER	34970	5896	2327	18
PMB	SemTag	67965	682	650	73

Table 4.2: Statistics about the datasets, reporting the number of sentences in train/dev/test set, and the number of labels.

Dataset	Example
ATIS	Do you have a $[United]_{Airline}$ flight from $[Boston]_{Origin}$
MIT Restaurant	I would like to find a $[Chinese\ buffet]_{Cuisine}$
MIT Movie	Did $[Sofia\ Coppola]_{Director}$ direct any adventure films
OntoNotes 5.0	A rescue team found $[Uchikoshi]_{Person}$ on $[the\ thirty\ first\ of\ last\ October]_{Date}$
PMB	$[Lucy]_{Person}$ $[can't]_{Negation}$ $[use]_{PresentSimple}$ $[chopsticks]_{Concept}$

Table 4.3: Example sentences for each dataset and its annotation

the implementation of Reimers and Gurevych (2017)[5]. The MTL models are trained in an alternate fashion (Jaech et al., 2016) between $\mathcal{T}_T$ and $\mathcal{T}_S$. Consequently, as the training data size of $\mathcal{T}_S$ is larger than $\mathcal{T}_T$, the same $\mathcal{T}_T$ data is reused until the whole $\mathcal{T}_S$ is used in the training. We evaluate the performance by computing the F1-score on the test set using the standard CoNLL-2000 evaluation[6].

4.4 Results and Discussion

Overall Performance. Table 4.4 lists the overall performance of the baselines and of the MTL models. We report the average F-1 score and also the standard deviation, as recommended by Reimers and Gurevych (2018), over three runs from different random

[5]https://github.com/UKPLab/emnlp2017-bilstm-cnn-crf
[6]https://www.clips.uantwerpen.be/conll2000/

Model	$\mathcal{T}_S$	$\mathcal{T}_\mathcal{T}$		
		ATIS	MIT-R	MIT-M
STL	-	$87.91_{0.56}$	$67.37_{0.26}$	$80.71_{0.63}$
STL+FB	-	$87.79_{0.67}$	$67.27_{0.64}$	$80.56_{0.54}$
MTL-FSN	*N*	$\mathbf{89.56}_{0.16}$	$68.82_{0.18}$	$80.77_{0.13}$
	S	$89.19_{0.26}$	$68.21_{0.71}$	$80.57_{0.32}$
	N,S	$89.10_{0.41}$	$68.21_{0.43}$	$79.69_{0.33}$
H-MTL	*N*	$89.17_{0.33}$	$\mathbf{69.22}_{1.00}$	$\mathbf{81.79}_{0.26}$
	S	$88.96_{0.41}$	$69.09_{0.24}$	$81.59_{0.17}$
	N,S	$88.78_{0.37}$	$68.96_{0.50}$	$81.15_{0.25}$

Table 4.4: Average F1-score and standard deviation (numbers in subscript) of the performance on the test sets. For the $\mathcal{T}_\mathcal{T}$ training split, only 10% data is used. **Bold** indicates the best score for each $\mathcal{T}_\mathcal{T}$. *N* and *S* in $\mathcal{T}_\mathcal{S}$ denote NER and SemTag, respectively.

seeds. For all $\mathcal{T}_T$, it is evident that the MTL models with NER or SemTag combinations yield the best results compared to STL. MTL models also outperform the STL + FB baseline, indicating that training the model simultaneously with the auxiliary task is better than incorporating the output of the independently trained auxiliary models as features for the slot filling model. In terms of the effectiveness of the auxiliary tasks, using NER produces the best results compared to the other $\mathcal{T}_S$ combinations. The difference between MTL with NER and MTL with SemTag is marginal, however, on all cases NER is more beneficial as an auxiliary task than SemTag. Combining both NER and SemTag as auxiliary tasks tends to perform worse than MTL with one auxiliary tasks. We hypothesize that for the model architecture that we use, it becomes more difficult to handle larger label set, for example, the number of label of SemTag is four times larger than NER and also the variation of text in SemTag is more diverse as the Parallel Meaning Bank corpus is constructed from twelve different sources.

Regarding the MTL models, in most cases, the performance of MTL-FSN and H-MTL are comparable. The most evident gap between H-MTL and MTL only shows on the MIT-M dataset in which H-MTL surpass corresponding MTL-FSN more than 1 F1

$\mathcal{T}_\mathcal{T}$	Concept	Model	
		STL	MTL
ATIS	LOC	$94.74_{0.37}$	$\mathbf{95.82}_{0.34}$
	ORG	$92.52_{0.89}$	$\mathbf{93.37}_{0.29}$
MIT-R	LOC	$75.29_{0.46}$	$\mathbf{76.02}_{0.39}$
MIT-M	PER	$\mathbf{85.04}_{0.24}$	$84.58_{0.56}$

Table 4.5: Performance on slots related to person (PER), location (LOC), and organization (ORG) concepts. We use the best MTL from Table 4.4 for each $\mathcal{T}_\mathcal{T}$.

points with relatively small standard deviation. Although on MIT-R, H-MTL obtains the best result i.e., 69.22 vs 68.22 from MTL-FSN, the standard deviation is relatively high on this specific case.

Slot-wise Performance. One of our motivations for using NER and SemTag is that their labels are coarse-grained, and that they can be re-used for several slot filling domains. We are interested to see whether MTL improves the performance of slots related to these coarse-grained concepts. In order to do this, we manually created a mapping[7] from the slots to some coarse-grained entity concepts used by CoNLL-2003 (Sang and Meulder, 2003) including Person, Organization, and Location. For example, in ATIS, the slot `airline_name` is mapped to Organization, the slot `fromloc.city_name` is mapped to Location, etc. We perform the analysis on the dev set by re-running the evaluation based on the mapping. Results in Table 4.5 show that in ATIS and MIT-R, MTL brings improvements on slots related to Location and Organization. However, MTL does not help in slots related to Person names in MIT-M. Based on our observation on the prediction results, most errors come from misclassifying `director` slots as `actor` slots. We sample 10 sentences in which the model makes mistakes on `director` tag. Of these sentences, four sentences are wrongly annotated. Another four sentences are errors by the model although the sentence seems easy, typically the model is confused between `director` and `actor`. The rests are difficult sentences. For example, the sentence: "*Can you name Akira Kurusawas first color film*". This sentence is somewhat general and the model needs more information perhaps access to an external knowledge base to discriminate between `actor`

[7]We provide the mapping in Table A.2 and A.3

and `director` to classify the entity "*Akira Kurusawas*".

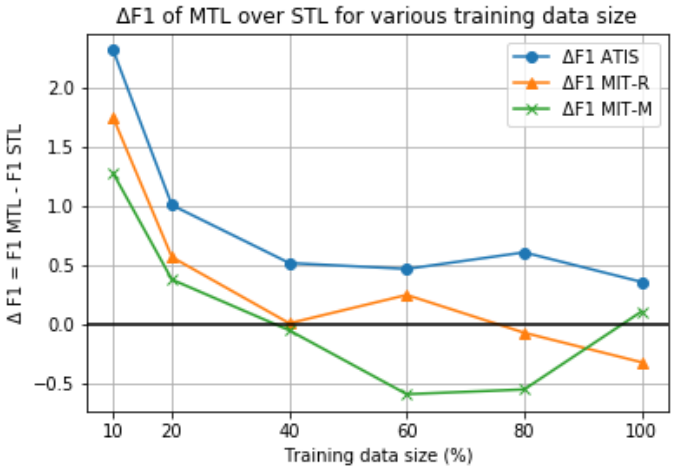


Figure 4.2: Gain ($\Delta F1$) obtained using MTL over STL on increasing training data. Positive numbers mean MTL is better, negative numbers mean MTL is worse. We use the best MTL from Table 4.4 for each $\mathcal{T}_\mathcal{T}$.

Performance Gain on Increasing Data Size. We also carried on an experiment by increasing the amount of training data on $\mathcal{T}_\mathcal{T}$, and evaluated the performance on the dev set to understand the usefulness of MTL on varying data size. As shown in Figure 4.2, as we increase the size of the training data, the gain that we obtain using MTL tends to decrease. The results suggest that MTL is indeed more useful in very low-resource scenarios, according to our initial hypothesis. After 40% training data size is used (around 2K utterances), MTL is less useful. We believe that this is because the slot filling datasets are relatively simple, e.g. the texts are short and most of them express a single specific request, thus, it is relatively easy for the model to capture the regularities.

Impact on Auxiliary Tasks Performance. We also perform an analysis to understand the effect of MTL to the model performance for $\mathcal{T}_\mathcal{S}$. The STL performance of OntoNotes and Semantic Tagging are around 89% and 96% respectively in terms of F1-score. With MTL, on average, the $\mathcal{T}_\mathcal{S}$ model performance decrease about 0.7 points for OntoNotes and 0.2 points for Semantic Tagging. This suggests that $\mathcal{T}_\mathcal{S}$ models do not benefit from the low-resource $\mathcal{T}_\mathcal{T}$ through the MTL framework and the training mechanism that we use. In general, whether MTL can benefit model performance in a target task given auxiliary tasks (or vice versa) is still a question and beyond the scope of this work. While there is

no exact answer yet for this question, we refer to (Bingel and Søgaard, 2017; Alonso and Plank, 2017) which study the characteristics of auxiliary tasks that is potential to help target task performance.

4.5 Data Selection

In the previous section we have shown that on the absence of conversational auxiliary text, non-conversational text as the source data may help low-resource NLU. We leverage non-conversational tasks, tasks that use widely available non-conversational text, through multi-task learning (MTL). While the results are positive in most cases, we seek a way to improve the MTL performance further by mitigating the distribution mismatch (Rosenstein, 2005) between the non-conversational source data and the conversational target data.

One solution to alleviate the impact of the mismatch is using data selection, a process for selecting relevant training instances from the source data. Data selection has been applied in the context of domain adaptation to address changes in the data distribution for various NLP tasks, such as sentiment analysis and POS Tagging (Ruder and Plank, 2017a; Liu et al., 2019a), machine translation (Axelrod et al., 2011) and Named Entity Recognition (NER) (Murthy et al., 2018; Zhao et al., 2018). To our knowledge, all existing previous works apply data selection to different domains, while maintaining the same task ($\mathcal{T}_\mathcal{S} = \mathcal{T}_\mathcal{T}$, $\mathcal{D}_\mathcal{S} \neq \mathcal{D}_\mathcal{T}$).

In our context, we aim to investigate the benefit of data selection in a more complex setting, where we have not only different domains ($\mathcal{D}_\mathcal{S} \neq \mathcal{D}_\mathcal{T}$), but also different tasks ($\mathcal{T}_\mathcal{S} \neq \mathcal{T}_\mathcal{T}$). Intuitively, such setting may bring advantage in situations where large training data are available for a task $\mathcal{T}_\mathcal{S}$, and we want to exploit such data for a different (although related) task $\mathcal{T}_\mathcal{T}$, where much less training is available. We experiment with the situation where $\mathcal{T}_\mathcal{S}$ is Named Entity Recognition (NER) on a general domain, where several datasets are available, and $\mathcal{T}_\mathcal{T}$ is slot filling in the context of utterance interpretation for dialogue systems, where much fewer data are available. Both of the tasks are poorly investigated in data selection (to our knowledge there is no previous work on slot filling), and there is no consensus about the benefit of data selection for them.

We propose an experimental framework where we can compare data selection settings with an increasing level of complexity. We first consider data selection where NER is both the source and target task, and apply transfer learning from different domains: we call this setting **S**ame **T**asks from **D**ifferent **D**omains (**STDD**) (§4.5.3), $\mathcal{T}_\mathcal{S} = \mathcal{T}_\mathcal{T}$ and $\mathcal{D}_\mathcal{S} \neq \mathcal{D}_\mathcal{T}$. In a second, more complex setting, we consider NER as the source task and slot filling as the target: this is called $\mathcal{T}_\mathcal{S} \neq \mathcal{T}_\mathcal{T}$ and $\mathcal{D}_\mathcal{S} \neq \mathcal{D}_\mathcal{T}$, **D**ifferent **T**asks from **D**ifferent **D**omains (**DTDD**) (§4.5.4). In this scenario, as we have disparate label space between the source and the target task, we combine the data selection process with multi-task learning (MTL). To our knowledge, this combination has received very little attention in the literature.

4.5.1 Framework

In general, the goal of data selection is to select an optimal subset of training instances, X_S^*, from all the available data X_S in $\mathcal{T}_\mathcal{S}$, to be used for training the model for the target task $\mathcal{M}_{\mathcal{T}_T}$. Given the source data $X_S = \{x_1^S, x_2^S, ..., x_n^S\}$, each instance is ranked according to a score $\mathcal{S}$ and the top m examples are then used to train $\mathcal{M}_{\mathcal{T}_T}$.

We apply the data selection approach from Ruder and Plank (2017b), based on Bayesian Optimization (BO) (Brochu et al., 2010), to evaluate the effectiveness of data selection on both the STDD and DTDD scenarios. Specifically, for DTDD we *combine* data selection and multi-task learning. Given X_S, the framework performs data selection based on a score $\mathcal{S}$ derived from a set of features. The top m examples are then used to train $\mathcal{M}_{\mathcal{T}_T}$. In case of STDD, the $\mathcal{M}_{\mathcal{T}_T}$ is a single task sequence tagging model, where we use a biLSTM-CRF model (§4.2.1). As for DTDD, $\mathcal{M}_{\mathcal{T}_T}$ is the MTL-FSN model (§4.2.2). The performance on the validation set of the target task is then used by the BO optimizer to update the weight of the scoring features. The overall framework is shown in Figure 4.3.

Following Ruder and Plank (2017b), the selection process is based on a score $\mathcal{S}$ computed as the linear combination of weighted features, which include both similarity and diversity features: $\mathcal{S}_\theta(x) = \theta^\top \cdot \phi(x)$, where θ represents the weight for each feature and $\phi(x)$ denotes the feature values of each instance x. The features are calculated between the representation of X_S instances and X_T. We use term distribution as the representation

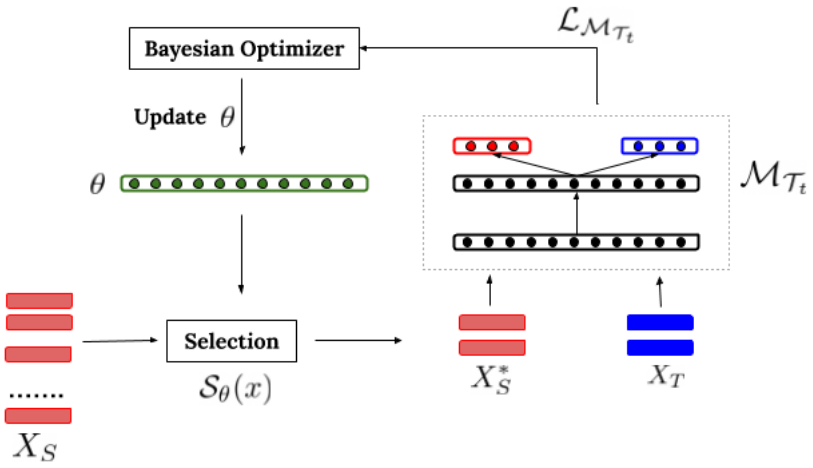


Figure 4.3: Overall Data Selection Framework

of the instances. We use the same similarity and diversity measures as Ruder and Plank (2017b). The weights θ are learned through BO by taking into account the performance on the validation set when selecting a particular subset of X_S. The score $\mathcal{S}$ is computed for each x in X_S, and then the top m examples are selected for training the $\mathcal{M}_{\mathcal{T}_T}$ model. The loss value $\mathcal{L}$ from the $\mathcal{M}_{\mathcal{T}_T}$ in the validation set is used by BO as a feedback to select the next points for θ.

4.5.2 Experiments

We systematically investigate how data selection is effective when applied on both the STDD and DTDD scenarios. We address two semantic sequence labeling tasks: Named Entity Recognition (NER) and slot filling (SF).

Datasets. For NER we use the OntoNotes 5.0 dataset that we use in MTL experiments (§4.3) but with additional sections. These sections include newswire (NW), talkshows broadcast (BC), telephone conversation (TC), news broadcast (BN), articles from web sources (WB), and articles from magazines (MZ). We use different OntoNotes sections as different domains in our experiments. The overall statistics of each section is shown in Table 4.6. As for SF dataset, we use the same datasets (Table 4.2) as our previous MTL experiments, namely ATIS, MIT-R, and MIT-M.

Data Selection Configurations. We make use of the selection framework described in Section 4.5.1, and apply three Bayesian Optimization data selection (*BODS*) configu-

Dataset	#train	#dev	#test	#label
OntoNotes NW	34970	5896	2327	18
OntoNotes BC	11879	2117	2211	18
OntoNotes TC	12891	1634	1366	18
OntoNotes BN	10683	1295	1357	18
OntoNotes WB	16598	2316	2307	18
OntoNotes MZ	6911	642	780	18

Table 4.6: Statistics about the NER datasets used in the experiments for data selection. The language of the datasets is English.

rations, according to whether we use features both for similarity and diversity ($DS_{sim,div}$), similarity features only (DS_{sim}), or diversity features only (DS_{div}). We compare the three configurations with the following baselines:

- All source, which uses all the data from $\mathcal{T}_S$.
- Random, which selects random data from $\mathcal{T}_S$.
- $DS_{map,full}$. We provide a manual mapping from NER labels to SF labels (Table A.2). A sentence from $\mathcal{T}_S$ is selected is if *all* the NER occurrences have a mapping to a slot in $\mathcal{T}_T$.
- $DS_{map,partial}$. A sentence from $\mathcal{T}_S$ is selected if *at least* one of the NER occurrences in the sentence has a mapping to a slot label in $\mathcal{T}_T$.

Training Settings. We follow most of the hyperparameters[8] as recommended by Reimers and Gurevych (2018). We train the model for $\mathcal{T}_S$ and $\mathcal{T}_T$ in an alternating fashion. We use early stopping on the dev set performance of $\mathcal{T}_T$. For the model performance evaluation, we calculate the F1-score using the standard CoNLL script[9]. For all experiments, we report the average F1 score results from 10 runs with different seeds. We follow Ruder and Plank (2017b) for most configurations of the optimizer (Table A.7), and run 50 iterations. For both the STDD and DTDD scenarios, we select top 50%[10] examples

[8] Appendix A.6 reports all used hyperparameters.
[9] https://www.clips.uantwerpen.be/conll2000.
[10] We tune from 10% to 50% on the dev set.

from X_S. For MTL we adapt the implementation from Reimers and Gurevych (2017), extending the Bayesian Optimization data selection framework from Ruder and Plank (2017b) to support MTL.

4.5.3 STDD Scenario: $\mathcal{T}_S = \mathcal{T}_T, \mathcal{D}_S \neq \mathcal{D}_T$

This scenario is the same setup as Ruder and Plank (2017b), where we use the same tasks both for the source and the target task from different domains, except that we apply the data selection to a semantic sequence tagging task namely NER. In this scenario, we use NER both for the source and the target task. The target domain is one of the three OntoNotes sections namely NW (news), TC (telephone conversation) and BC (mixed of conversation and broadcast) while as source domain ($\mathcal{D}_S$) we use all available sections in OntoNotes except the one used as the target domain. We only use 10% of training data for the target domain to simulate limited data settings. At the end of the data selection process, we select the top 50% sentences from $\mathcal{D}_S$ using the best feature weights learned with the Bayesian Optimizer.

Method	**TC**	**NW**	**BC**
Baseline			
All source	$\mathbf{63.17}_{4.75}$	$\mathbf{79.08}^{\dagger}_{0.42}$	$\underline{73.42}_{2.13}$
Random	$\underline{62.02}_{4.47}$	$77.93_{0.54}$	$71.39_{2.12}$
BODS			
$\text{DS}_{\text{sim,div}}$	$61.71_{4.57}$	$76.99_{0.40}$	$72.60_{1.14}$
DS_{sim}	$61.45_{3.80}$	$78.30_{0.41}$	$\mathbf{73.44_{1.12}}$
DS_{div}	$61.65_{3.77}$	$\underline{78.32}_{0.53}$	$71.89_{1.53}$

Table 4.7: Average F1-score and standard deviation on the test set. † indicates significant differences ($p < 0.05$) between the best BODS approach and the best baseline.

Table 4.7 compares the performance of the baselines with the selection-based approaches. In general, we do not observe clear advantages of data selection methods over the baselines, especially the all source data baseline. Using all source data yields the most competitive results almost in all cases. The only case in which DS surpasses the all source baseline is on the BC domain but only for a tiny gain. For NW and BC domains, some

DS methods show clear advantages over the random baseline, but still worse than using all source data.

We want to see whether the distance between domains may characterize the performance of the data selection. For this purpose we quantify the domain similarity between each pair $\mathcal{D}_\mathcal{S}$ and $\mathcal{D}_\mathcal{T}$ with Jensen Shannon Divergence (JSD) (Lin, 1991). We compute the JSD between the term distribution of $\mathcal{D}_\mathcal{S}$ and $\mathcal{D}_\mathcal{T}$. The average JSD of each target task with respect to the source tasks are 0.80 (TC), 0.86 (NW), and 0.87 (BC)[11]. We observe that the higher the JSD is, the more beneficial is the data selection for the target task. BC, which has the highest JSD average, benefits the most from the data selection. On the other hand, TC with the lowest average similarity, has the largest gap between the baseline and the best DS methods (-1.4 F1 point).

Based on our experiments, for the STDD scenario we observe that:

1. In most of the cases, DS methods are inferior to the all source baseline. Yet, each domain has a different selection metric configuration that performs the best. This observation suggests that the hypothesis from Ruder and Plank (2017b) i.e., different tasks or even different domains demand a different notion of selection metric, is also applicable to semantic sequence tagging tasks such as NER.

2. The gap between the best DS method and the baseline for each $\mathcal{D}_\mathcal{T}$ can be characterized from the average JSD similarity to its $\mathcal{D}_\mathcal{S}$. Being more similar to other $\mathcal{D}_\mathcal{S}$ is a more suitable situation to get benefit from data selection.

4.5.4 DTDD Scenario: $\mathcal{T}_\mathcal{S} \neq \mathcal{T}_\mathcal{T}, \mathcal{D}_\mathcal{S} \neq \mathcal{D}_\mathcal{T}$

In this scenario we intend to observe whether data selection adds benefit to MTL. As in the STDD case, data selection is performed on the auxiliary task, where data is assumed to be abundant, and we only use a small portion of data for the target task. We use NER as the auxiliary task and ST as the target task. Previously in §4.4, we show that NER is helpful for SF through MTL, although it is not clear whether adding data selection is beneficial. We use the MTL setup in §4.4 , where OntoNotes NW is used as the auxiliary

[11] Complete pairwise JSD values are listed in Table A.4.

Method	ATIS	MIT-R	MIT-M
STL			
biLSTM-CRF	$85.46_{0.25}$	$63.99_{0.77}$	$76.39_{0.57}$
Baseline (MTL)			
All source	$\mathbf{90.05_{0.34}}$	$69.28_{0.40}$	$81.28_{0.23}$
Random	$89.93_{0.26}$	$\mathbf{69.54_{0.35}}$	$\mathbf{81.35_{0.31}}$
$DS_{map,full}$	$89.97_{0.25}$	$68.82_{0.50}$	$79.27_{0.36}$
$DS_{map,partial}$	$89.85_{0.29}$	$69.24_{0.40}$	$80.76_{0.30}$
MTL+BODS			
$DS_{sim,div}$	$89.78_{0.39}$	$69.29_{0.37}$	$81.07_{0.29}$
DS_{sim}	$89.83_{0.31}$	$69.25_{0.41}$	$81.17_{0.25}$
DS_{div}	$89.95_{0.41}$	$69.09_{0.24}$	$81.10_{0.28}$

Table 4.8: Average F1-score and standard deviation on the test set. † indicates significant differences ($p < 0.05$) between the best BODS approach and the best baseline.

task, and the target task is one of the ST datasets with only 10% of available training data.

Observing the results in Table 4.8, in all the cases the baselines, namely all source data and random selection, perform better than MTL with DS methods. The selection methods based on manual label mapping, DS_{map}, do not bring advantage over all source data. Therefore, given two distant $\mathcal{D}_S$ and $\mathcal{D}_T$, selecting sentences based on the label mapping does not help. Moreover, as random selection gives good results as well for most scenarios, this indicates that data selection is not beneficial in our experimental setting that combines data selection and MTL.

Error Analysis. For the ATIS dataset, the models has trouble in disambiguating *flight origin* and *flight destination*. While for the MIT Restaurant dataset, similar type of problems occur between *cuisine*, *restaurant_name*, and *dish* slots. For the MIT-Movie dataset, misclassifications happen between *character*, *actor*, and *director* slots. We inspect the predictions made by the model to check what kind of cases are difficult for the model. For ATIS, there are several sentence patterns which are difficult for the model namely sentences that contain disjunction (*or*) or conjunction (*and*) when specifying the

flight origin or destination, sentences which has lack of context when specifying the slot information, and sentences which are very long. We list the example of these sentences in Table 4.9.

1	Anything from Baltimore **or** *Washington* with a stopover in Denver
2	Show me flights from Atlanta to Baltimore Denver **and** *Dallas*
3	Which airlines serves Denver Pittsburgh **and** *Atlanta*
4	Find me the earliest Boston departure **and** the latest *Atlanta* return trip so that i can be on the ground the maximum amount of time in Atlanta and return to Boston on the same day

Table 4.9: Example sentences from the ATIS dev set in which errors made by the model. (*highlighted*).

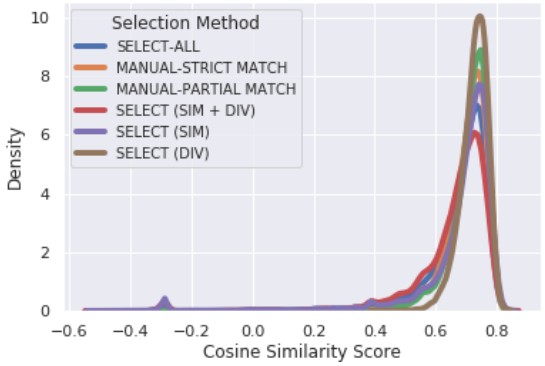
Our findings and lessons learned for DTDD are the following:

1. We observe that MTL performs better than single-task learning (STL) for low-resource slot filling, confirming the finding from Louvan and Magnini (2019). However, adding data selection for MTL is *ineffective* in our DTDD experimental setup. We hypothesize that MTL learns good common feature representations across tasks, this way inherently helping the model to focus on relevant features even from noisy data in $\mathcal{T}_S$. In addition to that, due to data sparsity in limited training, using all the training data works better because the model may learn a better text representation (sentence encoder). Recent similar work from Schröder and Biemann (2020) which uses information theoretic based for estimating the usefulness of an auxiliary task for MTL also found that for semantic sequence tagging tasks such as NER and argument mining, it is less clear when a particular dataset is useful as an auxiliary task.

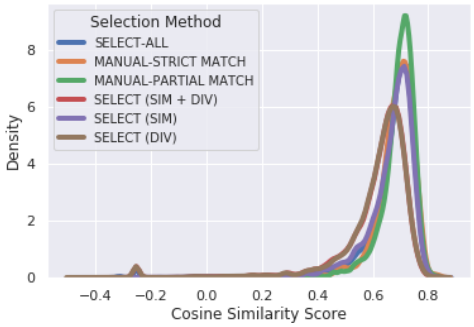
2. Data selection seems to produce selected sentences with concentrated similarity distribution. We analyze the similarity distribution of the sentences before and after the data selection (Figure 4.4). We use Infersent encoder (Conneau et al., 2017) to obtain the sentence embeddings and use cosine measure to calculate the similarity score. The similarity score is calculated between each sentence embedding

in $\mathcal{T}_S$ and the average values of sentence embedding in $\mathcal{T}_T$. We hypothesize that data selection is probably ineffective when the sentence similarity distribution between $\mathcal{T}_S$ and $\mathcal{T}_T$ is already concentrated on a very narrow range. However, when we perform data selection using several OntoNotes sections as $\mathcal{T}_S$ and perform further MTL experiment, the result is also negative (Appendix A.1).

3. Practically, using Bayesian Optimization for data selection is an expensive method especially when using it with multi-task learning.



(a) Similarity distribution from sentences selected from OntoNotes NW when ATIS is the target task



(b) Similarity distribution from sentences selected from OntoNotes NW when MIT-R is the target task

Figure 4.4: Sentence similarity distribution across different selection strategy

4.6 Conclusions

We proposed to leverage Named Entity Recognition and Semantic Tagging from non-conversational text as auxiliary tasks through multi-task learning to help low-resource slot filling. Our experiments demonstrate that non-conversational tasks are effective to improve slot filling performance, and they are reusable in different slot filling domains. We observe that incorporating a task-hierarchy in the multi-task architecture based on the granularity of the labels does not bring evident benefit in most of the datasets.

We also investigated the benefit of data selection for transfer learning in several scenarios of increasing complexity. We apply an existing model-agnostic state-of-the-art data selection framework, and carried on experiments on two semantic sequence tagging tasks, NER and slot filling, and two transfer learning scenarios, STDD (Same Tasks Different Domains), and DTDD (Different Tasks Different Domains). For the STDD scenario, selection methods show potential when the target domain has the highest similarity to the source domains, based on Jensen Shannon Divergence. As for the DTDD scenario in which we use related tasks (NER and Semantic Tagging) from distant domains, using selection does not bring advantage over using all the source data. A possible cause is that, because of data sparsity on the target task, it is only by injecting more source data that we can improve the model. Finally, MTL does not benefit from data selection, as it may already effectively help the model to focus on relevant features even though in the presence of noisy data from distant domains.

In a broader context, given several possible auxiliary tasks, it is still challenging to identify which auxiliary tasks can be useful for multi-task learning. There have been some works that study this issue, however it seems there is no strong consensus yet among these studies especially for semantic sequence tagging tasks.

Chapter 5

Generating Additional Labeled Data

As we have described in §3.6.1, in addition to transfer learning, another approach on alleviating limited labeled data is by applying *data augmentation*. Data augmentation approaches aim to generate additional labeled data automatically by transforming the original data through specific operations. Most of the data augmentation approaches in low-resource NLU are *model based* (Table 3.7), and we consider these approaches as *heavyweight augmentation* as they require to train a separate neural model and often involve several stages in the augmentation process.

In this chapter, we propose Lightweight Augmentation (Louvan and Magnini, 2020d) (contribution **C2**) methods that consist of relatively simple non-gradient based operations to produce utterance variations. Lightweight Augmentation includes meaning preserving slot value substitution and sentence modification through dependency tree manipulation. In addition to evaluating the effectiveness of Lighweight Augmentation on English datasets, we also assess the applicability of Lightweight Augmentation on five non-English datasets: Italian, Hindi, Turkish, Spanish, and Thai (Louvan and Magnini, 2020c) (§5.5).

5.1 Introduction

One of the methods proposed to alleviate data scarcity in task-oriented NLU is *data augmentation* (DA), which aims to automatically increase the size of the training data by

applying data transformations, ranging from simple word substitution to sentence generation. Recently, DA has shown promising potential for several NLP tasks, including text classification (Wei and Zou, 2019b; Wang and Yang, 2015), parsing (Sahin and Steedman, 2018; Vania et al., 2019a), and machine translation (Fadaee et al., 2017). As for SF and IC, DA approaches typically generate synthetic utterances by leveraging Seq2Seq (Hou et al., 2018a; Zhao et al., 2019; Kurata et al., 2016b), Conditional VAE (Yoo et al., 2019), or pre-trained Natural Language Generation (NLG) models (Peng et al., 2020). Such approaches make use of in-domain data, and are relatively *heavyweight*, as they require training neural models, which may involve several phases to generate, filter, and rank the produced augmented data, thus requiring more computation time. It is also relatively challenging for deep learning-based models to generate semantically preserving synthetic utterances in limited data settings.

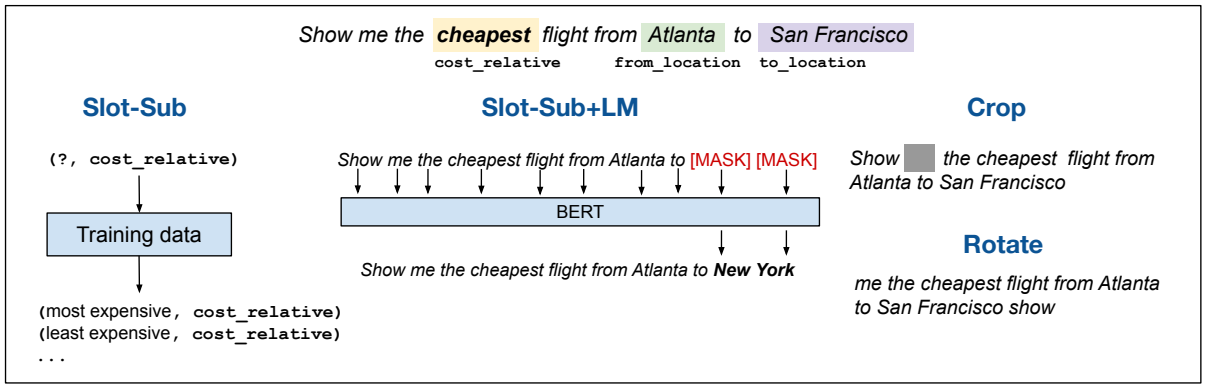


Figure 5.1: Examples of applying *lightweight augmentation* on an utterance in the ATIS dataset.

In this chapter, we show that *lightweight augmentation*, a set of simple DA methods that produce utterance variations, is very effective for SF and IC in a low-resource setting. Lightweight augmentation considers both *text span* and *sentence* variations. The span-level augmentation aims to diversify slot values in a particular text span through a *semantically preserving* substitution of slot values. The sentence-level augmentation seeks to produce alternative sentence structure through crop and rotate (Sahin and Steedman, 2018) operations based on a dependency parse structure.

We investigate the effect of lightweight augmentation both on typical biLSTM-based joint SF and IC models, and on large pre-trained LM transformers based models, in both

cases with a limited data setting. Our contributions are as follows:

- We present a lightweight text span and sentence level augmentation for SF and IC. We show that, despite its simplicity, lightweight augmentation is competitive with more complex, deep learning-based, augmentation.
- We show that big self-supervised models, such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), and ALBERT (Lan et al., 2020) can perform well under a low data regime, and still benefit from lightweight augmentation.
- The combination of our span based augmentation and transfer learning (e.g. BERT fine-tuning) yields the best performance for most cases.

5.2 Lightweight Data Augmentation

Given the original training data $\mathcal{D}$, DA aims to generate additional training data $\mathcal{D}'$. For each sentence S in $\mathcal{D}$, an augmentation operation is applied N times, which can be empirically determined. Each augmented sentence S' is added to $\mathcal{D}'$, and the union of $\mathcal{D}$ and $\mathcal{D}'$ is then used to train the model for SF and IC. We describe the Lightweight Augmentation operations in the following subsections.

5.2.1 Slot Substitution (Slot-Sub)

Our first lightweight method, slot substitution, is similar to (Gulordava et al., 2018), which is based on substituting a token in a sentence with another token with a consistent syntactic annotation (i.e., part-of-speech or morphology tags). However, unlike Gulordava et al. (2018), our method is not limited to single tokens. As slot filling is a *semantic* task, rather than syntactic, we can naturally extend the method from single tokens (i.e., slot names composed by a single token) to multiple tokens (i.e., slot names composed by multiple tokens, or *spans*[1]), still preserving the semantics associated to a certain slot.

Practically, for slot substitution we take advantage of the fact that SF training data

[1]We define a span as a sequence of one or more tokens that convey a slot value.

are typically annotated with the BIO format[2]. We exploit the fact that two text spans in different utterances in $\mathcal{D}$ are likely to be semantically similar if they share the same slot label. We randomly pick one span in the S and then perform the substitution (Figure 5.1 *Left*). For instance, we can substitute the span "*cheapest*", with other spans having the same slot label (i.e., COST_RELATIVE), such as "*least*" or "*most expensive*".

5.2.2 Slot Substitution with Language Model (Slot-Sub-LM)

Our second lightweight method, SLOT-SUB-LM, shares the goal with SLOT-SUB, i.e., to substitute sp with sp'. However, we do not use $\mathcal{D}$ to look for substitute candidates, instead we use a large pre-trained language model to generate the slot value candidates, using the *fill-in-the-blank* style (Donahue et al., 2020). The expectation is that large pre-trained LMs, being trained on massive amount of data, can produce a sensible text span given a particular sentence context, and possibly produce slot values that do not occur in $\mathcal{D}$. While we use BERT for our purpose, virtually any pre-trained LM can be used for SLOT-SUB-LM. Existing works on DA using LMs (Kobayashi, 2018; Kumar et al., 2020) are applied on text classification to replace random tokens in the text, which is not directly applicable to SF. Our approach focuses on *spans* conveying slot values, and include a filtering mechanism to reject retrieved slot spans that are not semantically compatible.

Generating New Slot Values. Given an utterance consisting of one or more slot value spans, we "blank" one of the span and then let the LM to predict the new tokens in the span. For instance, we give "*show me the* _____ *round trip flight from Atlanta to Denver*" to the LM for blank prediction. Practically, blank tokens are encoded as special `[MASK]` tokens[3] to let the pre-trained LM performing prediction. The decoding of the new tokens is carried out iteratively from left to right (Figure 5.1 *Middle*) and, to produce the surface form of a token, we apply nucleus sampling (Holtzman et al., 2020) using the top-p portion of the probability mass. Nucleus sampling has been empirically shown to be better than beam search, and top-k sampling (Fan et al., 2018) to produce fluent and diverse texts.

[2] `B` indicates the beginning of the span, `I` indicates the inside of the span. `O` indicates that a token does not belong to any slot. For example, "San Francisco" will be annotated as `B-to_location I-to_location`.

[3] We set the number of masked tokens to be the same as the tokens of the original slot value, e.g. `san francisco` is masked as `[MASK][MASK]`, although this number could actually be sampled as well.

Filtering. While pre-trained LMs are expected to generate sensible replacements for a span in the utterance, a possible issue is that the new slot span is not semantically consistent with the original one. For example, for the original span "*cheapest*" in "*show me the cheapest round trip flight from Atlanta to Denver*", the LM could output "*earliest*" as a substitution, which does not fit the slot label COST_RELATIVE. To mitigate this issue, we use a binary sentence classifier as a *filter* (**Slot-Sub-LM+Filter**) to decide whether S and S' are semantically compatible, based on the change made on the slot span. The training of the classifier is composed of a pair S and S', with its binary decision label (i.e., accept or reject S'). To construct the training data, for positive examples (*accept*) we take advantage of the sentence pair produced by SLOT-SUB, while for the negative examples (*reject*) we sample sp' in $\mathcal{D}$ where $y \neq y'$ and replace sp in S with sp' to produce S'. We use the BERT model as the sentence pair classifier and we encode the tokens, w, in both S and S' sentence pairs, as `[CLS]`$w_1^S w_2^S \dots w_n^S$ `[SEP]`$w_1^{S'} w_2^{S'} \dots w_m^{S'}$. On top of BERT, we add a feed-forward layer that uses the hidden state of the sentence representation, $h_{\texttt{[CLS]}}$, for prediction.

5.2.3 Crop and Rotate

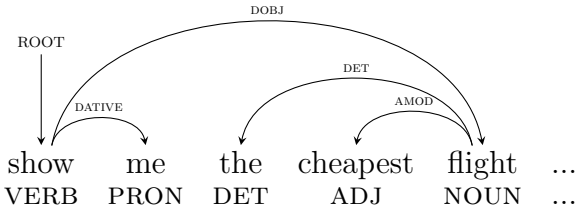


Figure 5.2: Original sentence.

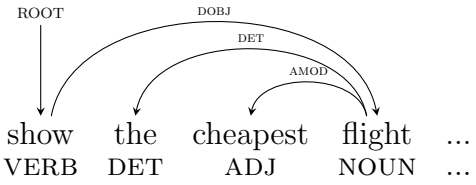


Figure 5.3: Sentence after applying CROP.

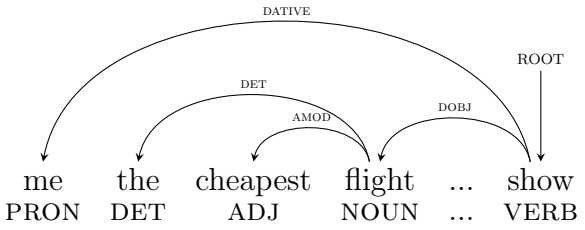


Figure 5.4: Sentence after applying ROTATE.

The third lightweight method that we present augments an utterance by changing its syntactic structure. We adopt the augmentation approach from Sahin and Steedman (2018), which is based on two operations, CROP and ROTATE applied to the dependency parse tree of a sentence. To our knowledge, this approach has not yet been applied to slot filling and intent classification, which is a contribution of our work. CROP focuses on particular fragments of a sentence (e.g., predicate and its subject, or predicate and its object), and removes the rest of the fragments, including its sub-tree, to create a smaller sentence. ROTATE aims to rotate the target fragment of a sentence around the root of the dependency parse structure, producing a new utterance. For example, in the utterance "*Show me the cheapest flight from Atlanta to San Francisco*", the word *"me"* can be cropped as it is one of the children of the *root* verb "*show*". While for rotation, the direct object ("*flight*") and its children ("*the cheapest*") are rotated around the root verb. Figure 5.3 and 5.4 illustrates the relevant dependency structure manipulation on a sentence (Figure 5.2).

It is possible for crop and rotate operations to produce ungrammatical or ill-formed sentences which can be considered as injecting noise during model training. It has been shown that noise induction as data augmentation can make a more robust model especially in limited data settings to reduce overfitting (Wei and Zou, 2019a; Li et al., 2017). In addition to that, having the variations from crop and rotate may beneficial for the model when applied to languages that have relatively flexible word order such as Turkish, Kazakh, and North Sami (Vania et al., 2019b; Sahin and Steedman, 2018).

5.3 Experiments and Results

We experimented our LIGHTWEIGHT AUGMENTATION approach on three well-known datasets for SF and IC, namely ATIS (Hemphill et al., 1990), SNIPS (Coucke et al., 2018) and FB (Schuster et al., 2019a). All datasets are in English. ATIS contains utterances related to flight domain (e.g., searching flight, booking). SNIPS includes multi-domain utterances such as weather, movie, restaurant, etc. FB contains utterances from 3 domains, weather, alarm, and reminder. To simulate the *data scarcity* setting, we follow previous works (Hou et al., 2018a; Yoo et al., 2019) and only use *medium*-size (i.e., 1/10) of training data for each dataset. Statistics on the three datasets are reported in Table 5.1.

	Label		**#Utterances** ($\mathcal{D}$)			**#Augmented Utterances** ($\mathcal{D}'$)			
Dataset	**#slot**	**#intent**	**#train**	**#dev**	**#test**	**SS**	**SS-LM**	**C**	**R**
ATIS	79	18	0.4K	500	893	3.9K	0.8K	0.8K	1.1K
SNIPS	39	7	1.3K	700	700	6.3K	2.5K	2.6K	3.7K
FB	11	12	3K	4.1K	8.6K	5.4K	5.4K	5.9K	8.5K

Table 5.1: Statistics of both the original training data $\mathcal{D}$ and the augmented data $\mathcal{D}'$. #train denotes our medium-size training data setup (10% of full training data). $\mathcal{D}'$ is produced by each augmentation method, where the number N of augmentations per sentence is tuned on the dev set. SS, SS-LM, C, and R denote SLOT-SUB, SLOT-SUB-LM, CROP, and ROTATE augmentation operations

As for evaluation, we use standard evaluation metrics, namely the F1-score for SF and accuracy for IC[4]. Performance is calculated as the average score of ten different runs. In order to compare our methods, we use two baselines for slot filling and intent detection: a simple BiLSTM-CRF model, and a state of the art BERT-based model, which is fine-tuned to SF and IC[5]. Each model is trained for 30 epochs, and we apply early stopping criteria.

For both slot substitution (SLOT-SUB) and slot substitution with language model (SLOT-SUB-LM) augmentation methods, we tune the number of augmented sentences

[4]Metric is computed using the standard evaluation script https://www.clips.uantwerpen.be/conll2000/

[5]We use the `bert-base-uncased` model

per utterance, N, on the dev set of each dataset. For crop and rotate, we use the default parameters from Sahin and Steedman (2018). To produce the dependency parse structure for the utterances in our datasets, we use Spacy[6]. All hyperparameters are tuned on the dev set. More details on the settings is provided in Table A.8. For training the binary classifier for SLOT-SUB-LM+Filter, we generate the same number of positive (*accept*) and negative (*reject*) training instances[7].

In order to allow comparison with more complex data augmentation approaches, we also report results obtained with state of the art approaches based on Seq2Seq (Hou et al., 2018a) and Conditional Variational Auto Encoder (CVAE) (Yoo et al., 2019). Our implementation is based on the Huggingface library (Wolf et al., 2019), available at `https://github.com/slouvan/saug`.

Table 5.2 reports the results on the test sets used in our experiments on limited data settings[8]. As for comparison, we include best-reported scores from two state of the art augmentation methods, namely a sequence-to-sequence (Seq2Seq) based on Hou et al. (2018a) and a VAE based methods from Yoo et al. (2019). Results in Table 5.6 (*test set*) show that lightweight augmentation is beneficial for both Bi-LSTM CRF and BERT, on both ATIS (single domain) and SNIPS (multi-domain) datasets. SLOT-SUB yields the best results for both the BiLSTM+CRF and BERT models, with SF performance up to 90.43 on ATIS and 90.66 on SNIPS, and IC performance to 95.49 on ATIS and 97.11 on SNIPS. As for the FB dataset, models only gain marginal improvement across lightweight augmentation. We hypothesize that FB is relatively easy to solve, compared with ATIS and SNIPS, as the slot filling performance of BiLSTM without augmentation already achieves a very high F1 score. The improvement using augmentation is more significant for SF rather than for IC.

Out of all LIGHTWEIGHT AUGMENTATION methods, SLOT-SUB obtains the best performance, particularly on slot filling on ATIS and SNIPS. The overall best performing configuration is a combination of BERT fine-tuning with SLOT-SUB augmentation. Given limited training data, BERT fine-tuning without augmentation surpasses BiLSTM-CRF without augmentation by a large margin. Yet, performance can be boosted even further

[6]https://spacy.io/

[7]Details about the training instance and the binary classifier performance is in Table A.9 and Table A.10 respectively

[8]For "upper bound" performance i.e., when 100% training data is used consult Appendix A.11

Model	DA	ATIS		SNIPS		FB	
		Slot	Intent	Slot	Intent	Slot	Intent
BiLSTM +CRF	None	86.83	90.64	84.51	95.94	93.83	98.47
	Seq2Seq (Hou et al., 2018a)	88.72	-	-	-	-	-
	VAE (Yoo et al., 2019)	89.27	90.95	-	-	-	-
	Slot-Sub	89.89†	93.37†	86.45†	96.30†	93.70	98.45
	Slot-Sub-LM	87.03	92.96†	82.82	96.14	91.52	98.20
	Slot-Sub-LM+Filter	87.19	92.01†	82.77	96.08	92.18	98.37
	Crop	88.62†	92.32†	85.84†	96.07	93.91	98.64
	Rotate	88.83†	92.33†	85.65	96.39†	94.04	98.56
BERT	None	89.39	94.98	89.17	96.70	94.22	98.61
	Slot-Sub	**90.43**†	**95.49**†	**90.66**†	**97.11**†	94.01	98.59
	Slot-Sub-LM	87.88	94.49	85.65	96.59	91.84	98.47
	Slot-Sub-LM+Filter	88.37	94.57	86.23	96.60	92.60	98.59
	Crop	89.47	94.55	89.77	96.78	94.20	98.73
	Rotate	89.57	94.48	89.37	96.81	**94.32**	**98.80**

Table 5.2: Overall results on the test set. Underlined numbers indicate best performing methods for a particular slot filling + intent model. **Bold** numbers indicate best overall methods. † indicates significant improvement over the baseline without augmentation (p-value < 0.05, Wilcoxon signed rank test).

with lightweight augmentation, suggesting that even a big, self-supervised model, such as BERT can still benefit from augmentation on limited data settings. The improvements on BiLSTM-CRF indicate that lightweight augmentation improves the model's robustness when trained on small amounts of data. We find that Slot-Sub-LM is suboptimal for SF. Our qualitative observation shows that Slot-Sub-LM often generates slot values that are semantically incompatible with the original slot label. Crop and Rotate can help IC in some cases, although their improvement is marginal.

Despite its simplicity, Slot-Sub is also competitive with state-of-the-art heavyweight

data augmentation approaches (Seq2Seq and CVAE), significantly boosting Bi-LSTM and BERT performance for SF on ATIS and SNIPS. We believe that the key advantage of Slot-Sub is its capability to maintain semantic consistency over the slot spans, which has revealed to be stronger than that of heavyweight approaches. This also shows that slot consistency is crucial for obtaining good performance, particularly for SF. While the CVAE based approach from Yoo et al. (2019) has injected slot and intent labels in the model, it seems that generating semantically consistent utterances is still challenging for deep learning models, especially when data is limited.

5.4 Analysis and Discussion

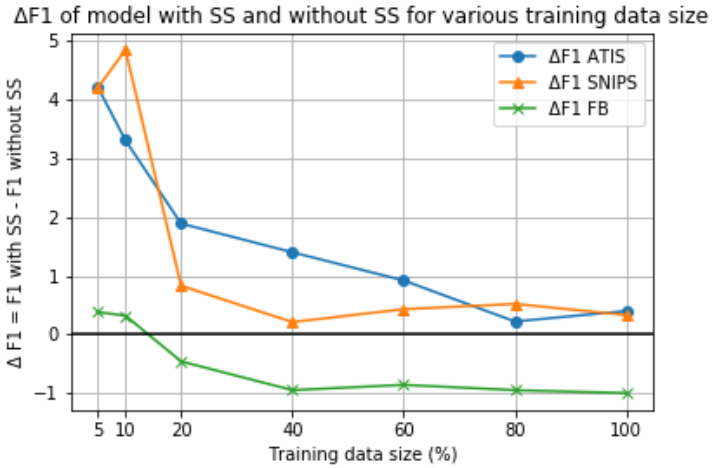


Figure 5.5: Gain ($\Delta F1$) obtained by Slot-Sub (SS) on various training data size. Positive numbers mean that the model with SS is better than without SS.

Performance on different training data size ($\mathcal{D}$). Figure 5.5 displays the gain obtained by Slot-Sub for various data size for slot filling. Using smaller data size (i.e., 5%) than our default setting, Slot-Sub still obtains a F1 gain for all datasets. On the other hand, as we increase the number of training data, the Slot-Sub benefit diminishes, without hurting performance on ATIS and SNIPS. As for FB we observe a performance drop of less than 1 F1, which is still relatively low.

Impact of number of augmented sentences. To better understand the effect of the number of augmented sentences per utterance (N), we now observe the performance of our

Model	Aug.	ATIS		SNIPS	
		Slot	Intent	Slot	Intent
BERT	None	91.6	95.0	89.8	95.0
(large)	SS	92.8	95.4	92.8	95.4
Albert	None	92.1	94.8	89.5	99.0
(xxl)	SS	92.9	95.0	**93.6**	**99.2**
Roberta	None	90.6	92.8	89.2	98.9
(large)	SS	**93.2**	**95.9**	92.5	98.8

Table 5.3: Lightweight augmentation SLOT-SUB (SS) applied to very large pre-trained LMs.

best performing method, SLOT-SUB, while changing N values (we use $\{2, 5, 10, 20, 25\}$) on the dev set[9]. As for ATIS, increasing N yields a F1 improvement from 90.68 up to 91.62; SNIPS performance increased from 87 F1 and to 88 F1 when increasing N from 2 to 5 and it is stable around 88 F1 when using N larger than 5; finally, FB is stable around 93.4 to 93.7 F1. Overall, the biggest improvement is when N is increased from 2 to 5, while with higher values only minor improvements can still be obtained on ATIS.

Is lightweight augmentation beneficial for very large language models? Motivated by the encouraging results that lightweight augmentation has obtained on a strong pre-trained LM such as BERT on low-resource settings (see Table 5.2), we now further examine the advantage of lightweight augmentation for other very large pre-trained LM models, namely Albert (Lan et al., 2020) and Roberta (Liu et al., 2019b). We use the largest trained models for each of the pre-trained LM, namely `bert-large-uncased`, `roberta-large`, and `albert-xxl`. Results, reported in Table 5.3, show that on limited data settings, all the very large models still benefit from SLOT-SUB, notably on the performance for SF.

Qualitative Analysis of slot values from Slot-Sub vs Slot-Sub-LM. The performance of SLOT-SUB especially in SF is better than SLOT-SUB-LM, as SLOT-SUB maintains semantic consistency on the span level. We observe that SLOT-SUB-LM often generates slot values that fit the sentence context but that do not maintain the semantics

[9]For the results on the dev set consult Table A.12

Dataset	Slot	Original Sentence	Slot-Sub	Slot-Sub-LM
ATIS	DEPART_TIME	List all flights leaving Denver on Continental on Sunday after **934 pm**	List all flights leaving Denver on Continental on Sunday after **7 pm**	List all flights leaving Denver on Continental on Sunday after **Christmas day**
	FROMLOC_CITYNAME	List all flights leaving **Denver** on Continental on Sunday after 934 pm	List all flights leaving **Atlanta** on Continental on Sunday after 934 pm	List all flights leaving **Boston** on Continental on Sunday after 934 pm
	AIRLINES_NAME	I need a flight on **Air Canada** from Toronto to San Diego with a layover in DC	I need a flight on **Northwest Airlines** from Toronto to San diego with a layover in DC	I need a flight on **a Thursday** from Toronto to San Diego with a layover in DC
SNIPS	CONDITION_DESCRIPTION	Will it be **sunny** in Eyota Hawaii on February seventh 2025	Will it be **humid** in Eyota Hawaii on February seventh 2025	Will it be **held** in Eyota Hawaii on February seventh 2025
	SPATIAL_RELATION	What is the **closest** cinema today playing animated movies	What is the **close-by** cinema today playing animated movies	What is the **underground** cinema today playing animated movies
FB	DATE_TIME	Set alarm **for 4 am** tomorrow morning	Set alarm **at 6** tomorrow morning	Set alarm **for me** tomorrow morning
	LOCATION	How hot is it in **Hong Kong** ?	How hot is it in **Fairbanks** ?	How hot is it in **the mornings** ?

Table 5.4: Samples of sentences from SLOT-SUB and SLOT-SUB-LM. The bold text span denotes the span that is substituted. The text span in blue denotes semantically consistent replacements, while red indicates semantically inconsistent substitutes.

of the slots, which hampers the performance in SF (Table 5.4). The fact that SLOT-SUB-LM often generates "wrong" slot values makes SLOT-SUB-LM+Filter also less effective. A possible future direction is to cast SLOT-SUB-LM as a *conditional* NLG problem, incorporating labels at the token-level, although this is still challenging when data is limited.

5.5 Follow-up Experiments: Non-English Datasets

In previous sections we have demonstrated the effectiveness of LIGHTWEIGHT AUGMENTATION on English datasets. Motivated by this encouraging results, in this section, we assess the applicability of LIGHTWEIGHT AUGMENTATION on non-English datasets.

Datasets. We experiment with datasets from five languages: Italian, Hindi, Turkish, Spanish, and Thai. For the Italian language, we use the data from Bellomaria et al. (2019), translated from the English SNIPS dataset (Coucke et al., 2018). SNIPS has been widely used for evaluating NLU models and consists of utterances in multiple domains. As for Hindi and Turkish, we use the ATIS dataset from Upadhyay et al. (2018), derived from Hemphill et al. (1990). ATIS is a well known NLU dataset on flight domain. As for Spanish and Thai we use the FB dataset from Schuster et al. (2019b) that contains utterances in alarm, weather, and reminder domains. The overall statistics of the datasets are shown in Table 5.5.

Baseline and Data Augmentation (DA) Methods. We use the state of the art BERT-based joint intent slot filling model (Chen et al., 2019) as the baseline model. We leverage the pre-trained *multilingual* BERT (M-BERT), which is trained on 104 languages. During training, M-BERT is fine tuned on the slot filling and intent classification tasks. Given a sentence representation $x = ([CLS]\, t_1\, t_2 \ldots\, t_L)$, we use the hidden state $h_{[CLS]}$ to predict the intent, and $h_{t_i}$to predict the slot label. As for DA methods, we did not include SLOT-SUB-LM. We add one configuration COMBINE, which combines the result of SLOT-SUB and ROTATE, as ROTATE obtains better results than CROP on the development set.

Results. The overall results reported in Table 5.6 show that applying DA improves performance on slot filling and intent classification across all languages. In particular, for SF, the SLOT-SUB method yields the best result, while for IC, ROTATE obtains better performance compared to CROP in most cases. These results are consistent with the

		#Label		#Utterances ($\mathcal{D}$)			#Aug ($\mathcal{D}'$)		
Dataset	**Lang**	**#slot**	**#intent**	**#train**	**#dev**	**#test**	**#SS**	**#C**	**#R**
SNIPS-IT	Italian	39	7	574	700	698	5.4K	1.4K	1.8K
ATIS-HI	Hindi	73	17	176	440	893	1.2K	460	472
ATIS-TR	Turkish	70	17	99	248	715	144	161	194
FB-ES	Spanish	11	12	361	1.9K	3K	1.4K	769	1K
FB-TH	Thai	8	10	215	1.2K	1.6K	781	-	-

Table 5.5: Statistics on the datasets. #train indicates our limited training data setup (*10% of full training data*). $\mathcal{D}'$ is produced by tuning the number of augmentations per utterance (N) on the dev set. SS, C, and R denote Slot-Sub, Crop, and Rotate augmentation operations

finding from our experiments on the English dataset (Section §5.3), where Slot-Sub improves SF and Crop or Rotate improve IC. In general, Rotate is better than Crop for most cases on IC, and we think this is because Crop may change the intent of the original sentence. Intents typically depend on the occurrence of specific slots, so when the cropped part is a slot-value, it may change the sentence's overall semantics.

We can see that languages with different typological features (e.g. subject/verb/object ordering)[10] benefit from Rotate operation for IC. This result suggests that augmentation can produce useful noise (regularization) for the model to alleviate overfitting when labeled data is limited. When we use Combine, it still helps the performance of both SF and IC, although the improvements are not as high as when only one of the augmentation method is applied. The only language that gets the benefits the most from Combine is Turkish. We hypothesize that as Turkish has a more flexible word order than the other languages it benefits the most when Rotate is performed.

Performance on varying data size. To better understand the effectiveness of Slot-Sub, we perform further analysis on different training data size (see Figure 5.6). Overall, we observe that as we increase the training size, the benefit of Slot-Sub is decreasing for all datasets. For some datasets, namely ATIS-HI and FB-ES, Slot-Sub can cause performance drop for larger data size, although it is reasonably small (less than 1 F1 point).

[10]Italian, Spanish, and Thai are SVO languages while Hindi and Turkish are SOV languages.

Model	DA	SNIPS-IT		ATIS-HI		ATIS-TR		FB-ES		FB-TH	
		Slot	Intent	Slot	Intent	Slot	Intent	Slot	Intent	Slot	Intent
mBERT	None	78.2	94.9	69.5	86.5	64.3	78.9	84.1	97.6	56.0	89.8
	SS	**81.9**†	94.9	**72.4**†	87.2	66.6†	79.8	**84.2**	97.7	**59.6**†	**91.4**†
	C	80.1†	94.6	70.0	86.9	65.1	79.4	83.8	98.0†	-	-
	R	79.2†	**95.3**	70.6	**87.6**†	65.2	80.0	83.2	**98.2**†	-	-
	Comb	81.2†	95.0	72.1†	86.9	**66.6**†	**81.1**†	83.6	97.9	-	-

Table 5.6: Performance comparison of the baseline and augmentation methods on the test set. F1 score is used for slot filling and accuracy for intent classification. Scores are the average of 10 different runs. † indicates statistically significant improvement over the baseline (p-value < 0.05 according to Wilcoxon signed rank test).

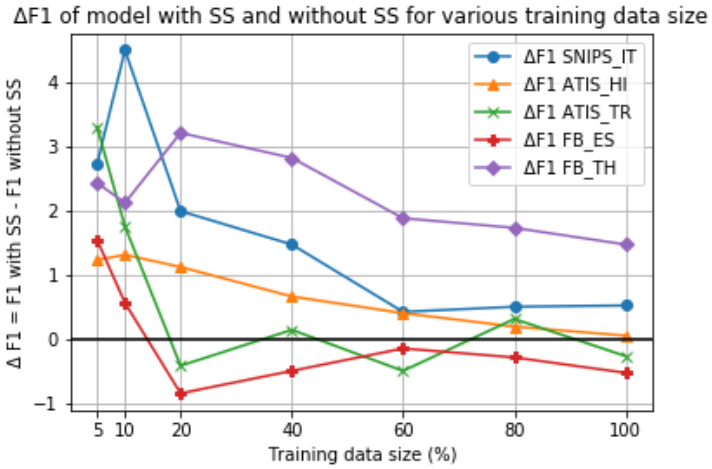


Figure 5.6: Improvement ($\Delta F1$) obtained by Slot-Sub (SS) on different training data size. Positive numbers mean that the model with SS yields gain.

FB-TH consistently benefits from Slot-Sub even when full training data is used. Until which training data size the improvement is significant vary across datasets[11]. For SNIPS-IT, improvement is clear for all training data size and they are statistically significant up until the training data size is 80%. For ATIS-HI improvements are significant until data size of 40%. As for FB datasets, improvements are significant only until the training data size is 10%. Overall, we can see that Slot-Sub is effective for cases where data is scarce

[11]For more details of the p-value of the statistical tests please refer to Table A.14

(5%, 10%), while it is still relatively robust for larger data size on all datasets.

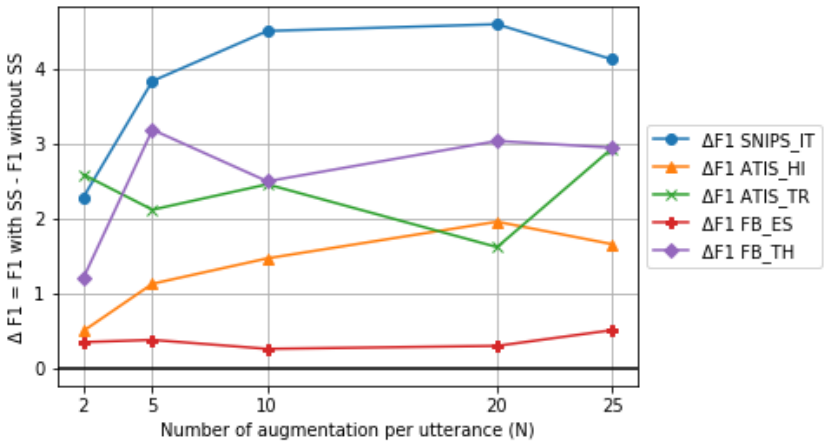


Figure 5.7: Gain ($\Delta F1$) obtained by SLOT-SUB (SS) on various number of augmented sentence (N). Positive numbers mean that the model with SS yields gain.

Performance on different numbers of augmentation per utterance (N). We examine the effect of a larger number of augmentations per utterance (N) to the model performance, specifically for SF (see Figure 5.7). For FB-ES, similarly to the results in Table 5.6, increasing N does not affect the performance. For the other datasets, increasing N brings performance improvement. For ATIS-HI, SNIPS-IT, and FB-TH the trend is that, as we increase N, performance goes up and plateau. For ATIS-TR, changing N does not really affect the gain of the performance as the performance trend is quite steady across number of augmentations. For most combinations of N in each dataset (except FB-ES), the difference between the performance of model that using SLOT-SUB and the model that does not use SLOT-SUB is significant[12].

5.6 Related Work

Data augmentation methods have been widely applied in computer vision, ranging from geometric transformations (Krizhevsky et al., 2012; Zhong et al., 2020), data mixing (Summers and Dinneen, 2019), to the use of generative models (Goodfellow et al., 2014) for generating synthetic data. Recently, data augmentation has been applied to various NLP

[12]For more details of the p-value of the statistical tests please refer to Table A.14

tasks, including text classification (Wei and Zou, 2019b; Wang and Yang, 2015), parsing (Sahin and Steedman, 2018; Vania et al., 2019a), and machine translation (Fadaee et al., 2017). Augmentation techniques for NLP tasks range from operations on tokens (e.g., substituting, deleting) (Wang and Yang, 2015; Kobayashi, 2018; Wei and Zou, 2019b), to manipulation of the sentence structure (Sahin and Steedman, 2018), to paraphrase-based augmentation (Callison-Burch et al., 2006).

Data augmentation has been also experimented in the context of slot filling and intent classification. Particularly, recent methods have focused on the application of generative models to produce synthetic utterances. Hou et al. (2018a) proposes a method that separates the utterance generation from the slot values realization. A sequence to sequence based model is used to generate utterances for a given intent with slot values placeholders (i.e., delexicalized), and then words in the training data that occur in similar contexts of the placeholder are inserted as the slot values. Zhao et al. (2019) also uses a sequence to sequence model by exploiting a small number of template exemplars. Yoo et al. (2019) proposes a solution based on Conditional Variational Auto Encoder (CVAE) to generate synthetic utterances. In this case the CVAE takes into account both the intent and the slot labels during training, and the model generates the surface form of the utterance, slot labels, and the intent label. Recent work from Peng et al. (2020) make use of GPT-2 (Radford et al., 2019), and fine-tuned it to intent and slot-value pairs to generate utterances.

In comparison to existing, state of the art, augmentation methods for slot filling and intent detection, the augmentation methods proposed in this work can be considered as *lightweight* because they do not require any separate training based on deep learning models for generating additional data. Still, lightweight augmentation maintains consistent slot semantic substitutions, a feature that is crucial for effective data augmentation. In the spectrum of existing augmentation methods, i.e., from words manipulation to paraphrasing-based methods, our lightweight approaches lie in the middle, as we focus either on particular *text spans* that convey slot values, or on particular structures in the dependency parse tree of the utterance.

5.7 Conclusion

We showed that LIGHTWEIGHT AUGMENTATION for slot filling and intent detection in low-resource settings is competitive to more complex deep learning based data augmentation. A lightweight method based on slot values substitution, while preserving the semantic consistency of slot labels, has proven to be more effective. We also show that large self-supervised models, like BERT, can benefit from LIGHTWEIGHT AUGMENTATION suggesting that a *combination* of data augmentation and transfer learning is very useful and has the potential to be applied to other NLP tasks.

Furthermore, we evaluate the effectiveness of LIGHTWEIGHT AUGMENTATION in five non-English typologically diverse languages: Italian, Hindi, Spanish, Turkish, and Thai. We find that the results are consistent with the experiment results on English datasets. Slot values substitutions and dependency tree manipulations can substantially improve performance in most cases when only a small amount of training data is available. We also show that a large pre-trained multilingual BERT (M-BERT) benefits from LIGHTWEIGHT AUGMENTATION.

In retrospect, there have been subsequent published works that align with our study. Dai and Adel (2020) evaluate a number of augmentation operations on Named Entity Recognition (NER) task on material science and biomedical domains. One such operation is mention replacement which is similar to our SLOT-SUB operation. Their experiments also show that relatively simple augmentation operations can boost performance for small training data scenario. Lin et al. (2021) use the BART pre-trained model (Lewis et al., 2020) which combines bi-directional encoder (BERT) and left to right decoder (GPT-2). Slot descriptions are injected during fine-tuning, and the loss function is modified to encourage more diverse slot values. Another important pre-training differences between BERT that we use for our SLOT-SUB-LM with BART is that for BART they use text-infilling task in which a text span is masked instead of a single token.

In a broader context, data augmentation has spawned a surge of interest in NLP recently. However, most data augmentation comes with empirical results and intuition, and it is still not clearly understood in the research community why data augmentation works (Feng et al., 2021).

Chapter 6

Continued Pre-Training for Zero-Shot Cross-Lingual SLU

As described in §1.2, most task-oriented Spoken Language Understanding (SLU) approaches focus on major languages, e.g., English, and it is still a challenge in SLU to achieve multilingual systems that support many languages. As the supervised learning model obtains the best performance in SLU, the bottleneck is to obtain a sufficient labeled dataset for many languages. Although we have experimented with several non-English languages in Chapter 5, collecting sizeable labeled data for many languages is infeasible, which drives most SLU research towards *cross-lingual* transfer approaches in which only a high-resource language, e.g., English labeled data is available for training the model and transfer directly to other languages (*zero-shot*).

The *de facto* method for zero-shot cross-lingual SLU consists of fine-tuning a pre-trained multilingual model on the English target task and then evaluating the model on unseen languages. However, recent studies show that adding a second pre-training stage (*continued pre-training*) can improve performance in certain settings, e.g. text classification tasks on English (Gururangan et al., 2020). This chapter corresponds to the contribution **C3**, in which we investigate the effectiveness of additional pre-training on intermediate unlabeled spoken language data in the context of zero-shot cross-lingual SLU.

6.1 Introduction

Many models have been proposed for SF and IC, and performance on standard English benchmarks have relatively saturated (Louvan and Magnini, 2020b). However, scaling models to other languages (*cross-lingual*), is still challenging, especially when labeled data is limited or not available (*zero-shot*). To address the problem, several zero-shot cross-lingual SLU approaches (Schuster et al., 2019a; Upadhyay et al., 2018; Xu et al., 2020) assume that a labeled dataset is available only for a high-resource language (e.g., English). With the rise of pre-trained multilingual language models, such as mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020), the default method for zero-shot cross-lingual SLU involves a *fine-tuning* stage: the pre-trained model is first trained on the English data with the task-specific objective, and then evaluated on the same task on languages that were not seen in the training phase (*zero-shot*).

However, while direct fine-tuning may serve as a strong baseline, pre-trained language models are not necessarily *universal*, and they may need domain-specific adaptation. In fact, recent works have shown that adding a second pre-training stage (*continued pre-training*) before fine-tuning can positively impact performance (Beltagy et al., 2019; Lee et al., 2020; Gururangan et al., 2020). In the continued pre-training stage, the pre-trained language model continues its training with a *domain-specific* or *task-specific* unlabeled dataset, using the same self-supervised objective (e.g., Masked Language Model). The motivation for adding another pre-training stage is to alleviate the *domain mismatch* – data distribution difference – between the original pre-training and the target task data. By performing continued pre-training on domain-specific unlabeled data, the model acquires prior knowledge expected to be helpful in the fine-tuning stage. While continued pre-training has shown promising results on text classification, typically on English, it remains unclear whether it is applicable in the context of *zero-shot cross-lingual* SLU.

In contrast to previous work, which has mostly focused on English text classification, we assess the effectiveness of continued pre-training for zero-shot cross-lingual SLU tasks (intent classification and slot filling). Our study reveals that the existing continued pre-training method (Gururangan et al., 2020), that is successful in English text classification tasks, does not always generalize to the context of zero-shot cross-lingual SLU.

We systematically investigate the effectiveness of continued pre-training of a pre-

trained multilingual model on zero-shot cross-lingual SLU for eight non-English languages, and analyzes the factors that crucially impact performance. Our study aims to answer the following questions:

(Q1) *Is continued pre-training effective for zero-shot cross-lingual SLU tasks?*
↪ Our experiments on the MultiATIS++ dataset (Xu et al., 2020) reveal that incorporating continued pre-training on intermediate English data can improve performance over direct fine-tuning for all languages either on slot filling on intent classification. The performance gain is especially evident for languages that use the Latin script writing system. The benefit of continued pre-training diminishes as we inject cross-lingual supervision in the fine-tuning stage, even with simple data augmentation through code-switching.
(Q2) *What are the factors that influence the effectiveness of the continued pre-training stage?*
↪ Using the target language for continued pre-training before fine-tuning on English introduces instability and can be detrimental to the overall performance; however, this instability can be largely alleviated with a simple code switch on the fine-tuning data. Regarding the continued pre-training data, we observe that performance improvement are not obtained by merely adding more data. In fact, higher domain similarity between the continued pre-training data and the fine-tuning data leads to better performance.

6.2 Background: Continued Pre-training

Many language models (Devlin et al., 2019; Liu et al., 2019b; Yang et al., 2019) are pre-trained with a self-supervised objective on massive unlabeled data from *general* domains (e.g., Wikipedia, Common Crawl) to acquire a powerful *contextual text representation*. They serve as a convenient initialization, and are then trained on the labeled data of downstream tasks (*fine-tuning*). However, the distribution shift between the original pre-training data and the domain of the target task can yield poor performance.

One method to mitigate the problem is to add a *continued pre-training* stage by continuing the pre-training of the model using the same self-supervised objective either on unlabeled domain-specific data (Lee et al., 2020; Beltagy et al., 2019; Gururangan et al., 2020) or on task-specific data (Gururangan et al., 2020). While continued pre-training has shown positive performance, most previous studies focus on text classification on

English. Our work is complementary: (i) we investigate the effectiveness of continued pre-training through intermediate data in a *zero-shot cross-lingual* setting; (ii) since our context is cross- lingual, we analyze the role that the *language* itself plays in the effectiveness of the continued pre-training; (iii) we investigate *both* sentence-level and token-level tasks, namely intent classification and slot filling, to be performed simultaneously on an utterance.

6.3 Continued Pre-Training in Zero-Shot SLU

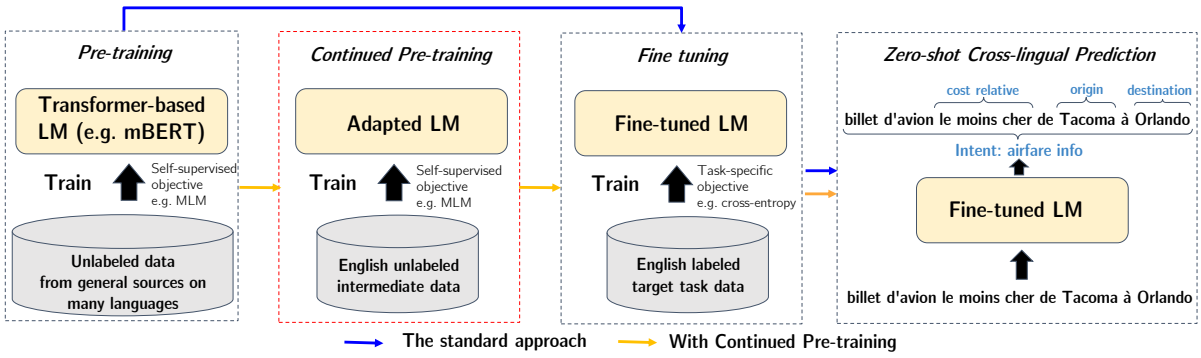


Figure 6.1: The overall stages of zero-shot cross lingual SLU using a pre-trained multilingual model. The standard approach follows the stages marked with blue arrows (*direct fine-tuning*). We investigate the effectiveness of adding a continued pre-training stage (red dashed box) in the overall pipeline.

The overall pipeline of zero-shot cross-lingual SLU is shown in Figure 6.1. We contrast the standard approach (direct fine-tuning) with the continued pre-training approach, which continues training the model on the intermediate *unlabeled data* with its original self-supervised objective, i.e., Masked Language Modeling (MLM). As the original pre-training data of many models are relatively far from the task-oriented dialogues used in SLU, we hypothesize that continued pre-training can alleviate the *domain* mismatch and add better prior knowledge before fine-tuning.

Intermediate Data for Continued Pre-Training. We define several criteria for the intermediate data that we use for continued pre-training. First, the domain of the intermediate data should be relatively close to the target dataset. We interpret the term domain as a multidimensional *variety space* (Ramponi and Plank, 2020; Plank, 2016):

a domain comprises multiple aspects – such as style, topic, and genre (van der Wees et al., 2015) – that contribute to text variation. In this perspective, and as the target dataset is SLU for a task-oriented dialogue system, we require that the intermediate data comprises text that presents a *spoken language dialog* style and covers a *broad range of topics*. Second, the dataset should be several magnitudes larger in size than the target task dataset. Third, it must be available in many languages to support our study of continued pre-training with the target language.

6.4 Experimental Setup

In this section, we detail the experimental settings related to models, evaluation metrics, and datasets.

6.4.1 Models

For all of our experiments, we use a transformer (Vaswani et al., 2017) based model, namely multilingual BERT (mBERT) (Devlin et al., 2019), as the pre-trained model. mBERT is originally pre-trained on Wikipedia from 104 languages with Masked Language Model (MLM) and Next Sentence Prediction (NSP) objectives. We use the pre-trained *bert-base-multilingual-cased*[1], consisting of 12 layers, 768 hidden states, 12 self-attention heads, and 110 M parameters. The input text is encoded with the format: *[CLS]* $w1, w2, w3, ..., w_n$ *[SEP]* where w_i is an individual token.

Continued Pre-training. For the continued pre-training stage, we train mBERT with unlabeled intermediate data only using the MLM objective for 12.5 K steps, and mostly adopt the hyperparameters[2] in Gururangan et al. (2020). We compare the following configurations: (i) DAPT_{Tgt} in which we perform continued domain adaptive pre-training (DAPT) of mBERT on intermediate unlabeled data on the target language. (ii) DAPT_{En} in which we perform continued domain adaptive pre-training of mBERT on intermediate unlabeled data on English.

Fine-Tuning. As baseline model, without any adaptation (No DAPT), we use the BERT-

[1] https://github.com/google-research/bert/blob/master/multilingual.md
[2] Appendix A.2

based joint intent and slot filling model proposed in (Chen et al., 2019). This model is state-of-the-art for IC and SF (Louvan and Magnini, 2020b), and it is often used as one of the baselines in recent zero-shot cross-lingual SLU studies (Xu et al., 2020; Li et al., 2021a). The hidden state $h^{[CLS]}$ is used for intent classification; the rest of the hidden states at each time step h_i are used for slot filling. The model is optimized with Adam, and the learning rate is set to 1×10^{-5}. The model is trained for 20 epochs on the English dataset; as the setup is zero-shot cross-lingual (i.e., no labeled target language data is available), we use the model on the last epoch for zero-shot evaluation following Xu et al. (2020).

We evaluate the effectiveness of each of the DAPT configurations when applied to the following fine-tuning scenarios:

- Fine-tuning on English (FINETUNE-EN). This is the standard fine-tuning scenario, where we take mBERT either with DAPT or no DAPT, fine-tune it to the English intent and slot filling data, and perform zero-shot prediction to all target languages.
- Fine-tuning on the English code-switched data (FINETUNE-CS). In this scenario, we perform data augmentation on the English fine-tuning dataset via code-switching. We follow the approach from Qin et al. (2020a) in which we replace English words with their translation in the target language using the Panlex bilingual dictionary (Kamholz et al., 2014). Given a training batch, we select which sentences and tokens will be replaced. We use the same hyperparameter[3] used by Qin et al. (2020a), that defines both sentence and word ratio to control the word replacement. We include FINETUNE-CS because we want to study the benefits of DAPT when adding stronger cross-lingual supervision in the fine-tuning stage.

We did not experiment with other models that leverage more informed code-switching (Liu et al., 2020c; Krishnan et al., 2021), or more complex models with machine translation and annotation projection (Xu et al., 2020), as our main goal is to investigate the effect of the intermediate data in the continued pre-training stage, rather than achieving the state of the art performance on the target task dataset.

Implementation & Model Evaluation metric. For IC and SF models we adapt the implementation from Qin et al. (2020a)[4]. For continued pre-training we use the script

[3] Appendix A.2

[4] https://github.com/kodenii/CoSDA-ML

from Wolf et al. (2019). For evaluation, we use accuracy for intent classification and entity-level F1 score for slot filling[5]. We run each experiment five times with different seeds and report the average accuracy and F1 score.

6.4.2 Dataset

Slot Filling and Intent Classification Dataset. We use the MultiATIS++ (Xu et al., 2020) dataset, which contains nine languages: English, German, Spanish, French, Portuguese, Hindi, Japanese, Chinese, and Turkish. The dataset is derived from the original ATIS English dataset (Hemphill et al., 1990), widely used as a benchmark for intent classification and slot filling for task-oriented dialogue systems. Utterances are related to conversations of a user asking for flight information to a system and are annotated with their intent and slots in BIO format (Ramshaw and Marcus, 1995b). The overall dataset statistics are shown in Table 6.1[6].

Language	#train	#dev	#test	#slot	#intent
English	4.4K	490	893	83	24
German	4.4K	490	892	83	24
Spanish	4.4K	490	893	83	24
French	4.4K	490	893	83	24
Portuguese	4.4K	489	892	83	24
Hindi	1.4K	160	888	74	22
Japanese	4.4K	490	886	83	24
Chinese	4.4K	490	893	83	24
Turkish	0.6K	60	715	70	21

Table 6.1: Multi-ATIS++ (Xu et al., 2020) dataset statistics.

Continued Pre-training Dataset. In this work, we focus on using the OpenSubtitle[7] (OpenSub) (Lison and Tiedemann, 2016) dataset for the continued pre-training stage for

[5]We use the standard CoNLL script to compute the F1 score http://deeplearning.net/tutorial/code/conlleval.pl

[6]We found slot annotation mistakes, especially for Spanish and French datasets, where the number of tokens and the corresponding slot labels are different. For these cases, we skip the utterances.

[7]https://opus.nlpl.eu/OpenSubtitles-v2018.php

several reasons. First, the dataset is constructed from movies and TV series containing *spoken language* in dialogue settings covering a broad range of topics. Second, OpenSubtitle covers all the *languages* that we use on the downstream tasks, which enables us to evaluate not only DAPT_{En} but also DAPT_{Tgt}. Third, the dataset is large in size, thus ideal for continued pre-training. Typically, the dataset used for continued pre-training is larger than that used for fine-tuning. For our experiments we randomly sampled 100K sentences for each language in the OpenSub dataset, resulting in a dataset around 20 times larger than the downstream task dataset. Table 6.2 summarizes the statistics for the OpenSub dataset.

Language	**Total Tokens**
English (EN)	734,302
German (DE)	691,039
Spanish (ES)	711,264
French (FR)	739,551
Portuguese (PT)	676,789
Hindi (HI)	688,675
Japanese (JA)	747,780
Chinese (ZH)	611,700
Turkish (TR)	554,709

Table 6.2: OpenSub (Lison and Tiedemann, 2016) dataset statistics. Each language has 100K utterances.

6.5 Results

The main goal of our experiment is to answer research question (**Q1**) *Is continued pre-training effective for zero-shot cross-lingual SLU tasks?* Table 6.3 compares the zero-shot performance for slot filling and intent classification across languages. Observing the results language-wise (by column in Table 6.3), all languages improve over No-DAPT in at least one DAPT setting. Hence, DAPT is effective across languages. Note that the baseline performance, No-DAPT, on FineTune-En scenario is lower from FineTune-CS because FineTune-CS has stronger supervision signal on the target language i.e.,

Slot filling F1								
	DE	ES	FR	PT	HI	JA	ZH	TR
FineTune-En								
No-DAPT	65.3	71.3	64.0	61.9	47.5	62.2	66.3	27.4
$\Delta\text{DAPT}_{\text{Tgt}}$	+4.0	−2.4	−7.7	−0.6	−12.9	−9.7	−0.6	+18.5
$\Delta\text{DAPT}_{\text{En}}$	+2.1	+0.9	+5.9	+1.4	−4.5	+0.8	−0.2	−5.8
FineTune-CS								
No-DAPT	75.5	80.8	71.9	72.0	58.1	67.1	81.6	72.0
$\Delta\text{DAPT}_{\text{Tgt}}$	−0.2	−0.4	+0.5	+1.1	−3.9	−6.3	−1.2	−10.9
$\Delta\text{DAPT}_{\text{En}}$	+0.4	+0.1	+4.6	+1.2	−13.9	−8.4	−0.7	−15.8

Intent classification accuracy								
	DE	ES	FR	PT	HI	JA	ZH	TR
FineTune-En								
No-DAPT	90.0	91.9	92.1	92.8	81.1	83.0	87.1	61.2
$\Delta\text{DAPT}_{\text{Tgt}}$	−10.8	+0.5	−13.3	−1.6	−13.3	−1.9	−2.9	+8.1
$\Delta\text{DAPT}_{\text{En}}$	−0.8	−0.1	+0.1	−0.6	−2.5	−0.5	−2.4	+8.3
FineTune-CS								
No DAPT	95.1	96.4	96.6	94.2	85.6	85.1	88.0	66.2
$\Delta\text{DAPT}_{\text{Tgt}}$	−1.1	−0.2	−0.5	+1.3	+0.6	−2.4	+0.3	+3.9
$\Delta\text{DAPT}_{\text{En}}$	−1.6	−0.2	−0.2	+0.4	−0.8	−2.6	−7.3	+12.3

Table 6.3: Performance comparison on the test set for slot filling and intent classification. Scores for No DAPT are the average slot F1 and intent accuracy from five runs. The $\Delta\text{DAPT}_{\text{Tgt}}$ and $\Delta\text{DAPT}_{\text{En}}$ indicate the delta between DAPT and No DAPT.

the English fine-tune dataset is code-switched to the target language. Although the pre-trained multilingual language model is pre-trained on hundreds of languages, the context for training is monolingual. Code-switching during fine-tuning helps to improve the cross-lingual representation as it mixes context from different languages. These baseline results

also confirm the finding from Qin et al. (2020a) that code-switching boosts zero-shot cross-lingual performance for slot filling and intent classification.

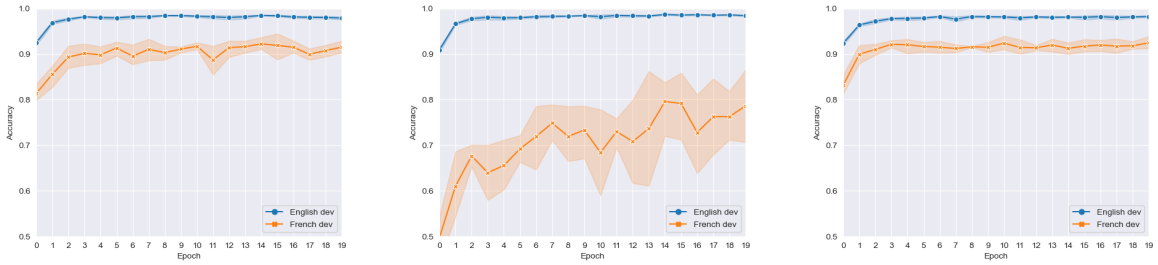
Observing the results per task, slot filling benefits from either DAPT_{En} or DAPT_{Tgt} for German, Spanish, French, Portuguese, and Turkish – *languages with Latin scripts writing systems*. For these languages, the margin obtained from DAPT when fine-tuning on English (FineTune-En) is higher than when we apply DAPT on code-switched data (FineTune-CS). The margin of DAPT when applied on FineTune-CS diminishes because FineTune-CS uses a stronger supervision signal in the fine-tuning stage, thus providing a higher baseline. For non-Latin script languages, performing continued pre-training is less useful, and we only observe marginal improvement on Japanese when applying DAPT_{En} and FineTune-En. Similar to Lauscher et al. (2020), we believe that performance is also affected by typological language proximity such as the subject, verb, and object ordering, phonology features or other aspect related to the original size of the pre-training data of mBERT. We leave this for future work.

DAPT is less effective for intent classification than for slot filling. The only language that consistently benefits from continued pre-training in both fine-tuning scenarios is Turkish. In general, it is harder to boost Latin script languages' performance through DAPT because the baseline is relatively high: a stronger supervision signal would thus be needed. The performance gain is small even for those languages that do benefit from DAPT. We also observe that using a different language between continued pre-training and fine-tuning stages, DAPT_{Tgt} and FineTune-En, may hamper performance even for Latin script languages, especially Spanish, French and Portuguese. We further discuss the effect of using different language between continued-pretraining and fine-tuning in Section §6.6.1.

6.6 Analysis and Discussion

To answer research question (**Q2**) *What are the factors that influence the effectiveness of the continued pre-training stage?* we analyze our results focusing on the performance variation when using different languages in DAPT and fine-tuning (§6.6.1), the effect of domain distribution in different sources for DAPT_{En} (§6.6.2). In addition to that, we perform performance analysis on per slot and per intent basis (§6.6.3).

6.6.1 Performance variation when applying DAPT



(a) No DAPT + FineTune-En (b) DAPT_{Tgt}+ FineTune-En (c) DAPT_{En}+ FineTune-En

Figure 6.2: Post-hoc analysis: *development set* performance variation (with a 95% confidence interval) on intent classification between English and French, using FineTune-En and applying different DAPT strategies.

Table 6.3 shows that DAPT is effective across languages. In particular, DAPT_{En} is superior to DAPT_{Tgt} for continued pre-training. However, as we have noticed in Section §6.5, there are cases where performance drop when we use DAPT_{Tgt} and FineTune-En, especially for intent classification. This behaviour holds even for languages relatively close to English, such as German and French. One possible reason for the drop in accuracy is that the language difference introduces instability in fine-tuning. Our post-hoc analysis shows that the target language performance during training on the dev set has a large deviation and continues fluctuating even after the English dev performance has stabilized. This observation resonates with a previous study from Keung et al. (2020), which shows that, for zero-shot text classification, English dev performance often does not correlate with those of the target language. Using DAPT_{Tgt} and FineTune-En pronounces the disagreement of performance between the English and the target dev set. Figure 6.2 shows the comparison of the intent classification performance during training across continued pre-training strategies when fine-tuning on English for French[8]. However, for the slot filling task, we do not observe a large performance variation even with a language mismatch: this might indicate that text classification is more susceptible to instability than sequence tagging. The variability caused by DAPT_{Tgt} is largely alleviated when we use DAPT_{En}. For the FineTune-CS scenario, the system is relatively stable even when combined with

[8]For other languages, refer to Figure A.1

DAPT_{Tgt} or DAPT_{En}.

6.6.2 Domain Relevance for DAPT_{En}

We aim at investigating whether the improvement from the continue pre-training derives indeed from the domain relevance of the intermediate data. For this purpose, we selected a few *written text* datasets instead of spoken language, which are focused on a *specific topic*.

Specifically, we use the European Medicines Agency[9] (EMEA) and European Central Bank[10] corpus (ECB) from Tiedemann (2012). EMEA contains articles about human, veterinary, or herbal medicines extracted from the EMEA website. ECB contains financial documents that are extracted from the website and documentation of the European Central Bank. In order to check that EMEA and ECB are more distant in terms of domain from MultiATIS than OpenSub, we compute the Jensen Shannon Divergence (JSD) measure of the term distribution. JSD is often used to measure the domain similarity between two datasets (Dai et al., 2020; Ruder and Plank, 2017b). We compute the JSD between the MultiATIS English dataset that is used for fine-tuning and each English intermediate dataset (Table 6.4). Based on the domain similarity measure, EMEA and ECB are more distant to MultiATIS than OpenSub.

	OpenSub	EMEA	ECB
JSD	0.419	0.391	0.397

Table 6.4: Domain similarity between MultiATIS and each of the intermediate data.

For each intermediate dataset, we randomly sample 100K sentences. We compare the slot filling performance of DAPT_{En} with FineTune-En on OpenSub, EMEA, and ECB in Table 6.5. Overall, we see that the DAPT using OpenSub obtains improvements over No-DAPT in all cases. The DAPT performance using EMEA and ECB are lower than OpenSub in most cases. Even for DE and PT languages, DAPT with ECB obtains substantially lower performance than No-DAPT. However, there are cases when EMEA or

[9] https://opus.nlpl.eu/EMEA.php
[10] https://opus.nlpl.eu/ECB.php

Lang.	No DAPT	ΔDAPT$_{\text{En}}$		
		OpenSub	EMEA	ECB
DE	65.3	+2.1	−2.5	−9.5
ES	71.3	+0.9	+0.9	+1.3
FR	64.0	+5.9	+2.0	+0.7
PT	61.9	+1.4	−0.3	−9.1
Avg		+2.5	+0.005	−4.1

Table 6.5: Comparison of slot filling (top) performance by applying DAPT$_{\text{En}}$ with FineTune-En using OpenSub, EMEA, and ECB.

OpenSub
You have a cancellation on Flight 16 for New York .
That route does not take us to the airport .
Chicago , this is flight 209er
I fly to Taiwan Tuesday then back to Dusseldorf

Table 6.6: Example of the most similar sentences from OpenSub to the utterance in MultiATIS: *Show me flights from Denver to Philadelphia on a Monday*

.

ECB match or even perform better than OpenSub i.e., for Spanish language. These cases indicate that performing *data selection* before continued pre-training could be beneficial for constructing more optimal DAPT dataset. We leave this possibility for future work.

Qualitative Analysis. We hypothesize that OpenSub contains some domain-relevant utterances with MultiATIS that may support positive transfer in DAPT. We retrieve an utterance from MultiATIS, "*Show me flights from Denver to Philadelphia on a Monday*" that is close to the centroid[11] of the MultiATIS dataset. Then, we retrieve the most similar utterances from OpenSub using cosine similarity of the BERT-based sentence representation (Table 6.6). These utterances are not directly related to booking flight tickets but relatively relevant to flight or travel topics in general. Another evident dimension

[11]This is computed by averaging the BERT based sentence representation

in these similar utterances[12] that may help positive transfer in DAPT is the style of the utterance, which is mainly in a conversational style.

6.6.3 Per Slot and Intent Breakdown

We perform further analysis on the development set of the target language to examine the performance on a per-slot basis. We focus our analysis on the FINETUNE-CS scenario. As the number of slots is relatively large, we categorize each slot into few coarse-grained types: for example, slots such as FROMLOC.CITY_NAME and TOLOC.CITY_NAME are mapped to LOCATION (LOC), AIRLINE_NAMES to ORGANIZATION (ORG), DEPART_TIME.PERIOD_OF_DAY, DEPART_TIME.TIME etc to TIME. In general, slots that belong to location, date and time, account for around 80% of the overall slots. We also include flight-specific slots (e.g., FLIGHTMOD) in this analysis, which describe the criteria for choosing a flight; whether it is the earliest, the shortest in terms of duration, etc. As shown in Figure 6.3, slots that belong to *location* benefit from DAPT for all European and Turkish languages. In terms of entity type improvement, German is similar to French: both languages improve on the date and time entity types; Spanish and Portuguese are also similar since they improve on the *organization* type. For non-Latin script languages, although in terms of overall results (Table 6.3) DAPT does not improve performance, we see improvements on slots related to date (for Hindi), location (for Chinese), and also slots related to flight-specific information (for Japanese). We conduct the same type of analysis on the dev set for intent classification (Figure 6.4) when applying DAPT_{En} to FINETUNE-CS. We focus on intents that have frequency > 20. On these intents, the margin between DAPT_{En} and no DAPT is tiny; this is expected as the overall performance between DAPT_{En} and no DAPT for intent classification is comparable. In addition to that, the intent *atis_flight* occupies 75% of the data, and its performance is already high (around 98 - 99% for Latin script languages) even without DAPT.

Qualitative error analysis. We perform error analysis on the French dev set. Most errors made by the No DAPT model that are fixed by DAPT_{En} model belongs to three slots: FROMLOC.CITY_NAME, TOLOC.CITY_NAME, and STOPLOC.CITY_NAME. We manually examine 92 examples of these errors. Most of the errors that are fixed by DAPT_{En} are the cases in which the No DAPT predicts the gold slot value as `O` (*false negative*).

[12] More examples in Appendix A.15

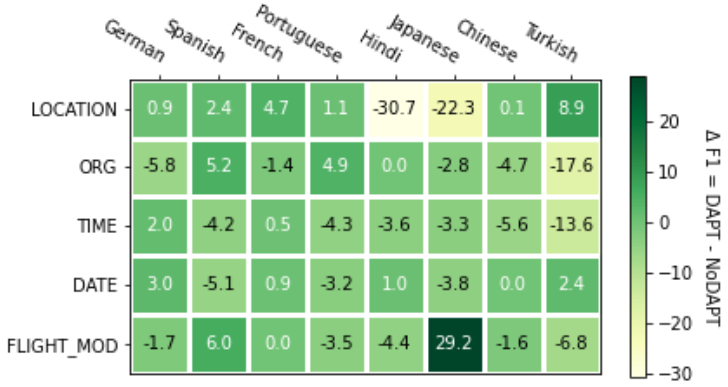


Figure 6.3: Δ F1 performance on per slot basis between the DAPT_{En} and no DAPT on each language on FineTune-CS scenario. Positive value means DAPT yields performance gain over no DAPT.

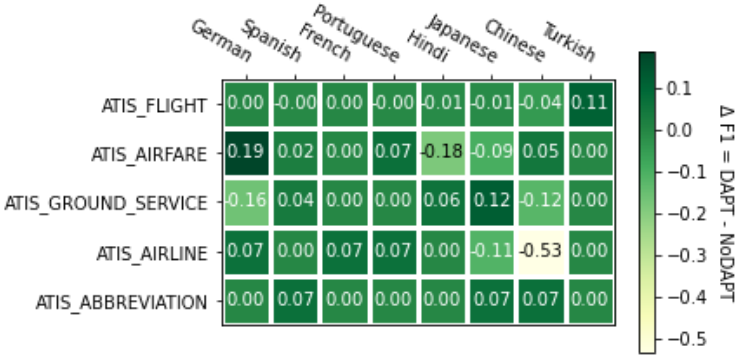


Figure 6.4: Δ accuracy on per intent basis between the DAPT_{En} and no DAPT on each language on FineTune-CS scenario.

For these cases, we found that the model has problems in recognizing a slot value when it is preceded by a partitive article *d'*, for example "*Vols d' Atlanta le deux août* (*Flights from Atlanta Aug. 2*)". Another type of error is related to ambiguity: for example, a FROMLOC.CITY_NAME slot is predicted as TOLOC.CITY_NAME. Furthermore, the error cases in which both models have difficulties are related to *partial match errors* which often involve consecutive tokens with different slot types. For example, in the utterance "*Je veux voyager de Washington DC à Philadelphie mardi matin*", both models predict *Washington DC* as the city name, while Washington and DC are annotated as separate slots, representing the city name and the state name respectively.

6.7 Related Work

Zero-Shot Cross-Lingual SLU. Before the advent of the pre-trained multilingual transformer models, most approaches relied on pre-trained cross-lingual embeddings to perform zero-shot SLU. Upadhyay et al. (2018) uses cross-lingual embedding (Bojanowski et al., 2017) to perform zero-shot SLU while Schuster et al. (2019a) uses multilingual embedding (Cove) from pre-trained multilingual bi-LSTM encoder used in Neural Machine Translation (NMT). Liu et al. (2019c, 2020c) leverages transferable latent variables to improve the sentence representation across languages. More recently, as pre-trained multilingual transformer models show potential in zero-shot settings, most approaches focus on improving their multilingual representation through augmentation and alignment methods. Qin et al. (2020a) proposes multilingual code-switching using a bi-lingual dictionary to improve mBERT's multilingual representation. Xu et al. (2020) introduces soft alignment of slots between English and the target language produced by an NMT system that eliminates the need for an annotation projection pipeline. Kulshreshtha et al. (2020) study the effect of various cross-lingual alignment methods to improve mBERT representation.

Continued Pre-training. Domain adaptation is a long-studied problem in the NLP community (Daumé III, 2007; Blitzer et al., 2007), in which we assume data in the target domain might be hard to obtain while being abundant in source domains. Continued pre-training – where the model is trained on relevant data using the same pre-training objective – is used for mitigating the distribution mismatch between the pre-training and the fine-tuning data in terms of *domain* (Logeswaran et al., 2019; Han and Eisenstein, 2019; Gururangan et al., 2020; Beltagy et al., 2019), *task* (Gururangan et al., 2020), and *language* (Pfeiffer et al., 2020). A complementary approach performs a first fine-tuning on related auxiliary tasks (for which training data are easy to obtain) before the final fine-tuning on the downstream task (Arase and Tsujii, 2019; Garg et al., 2020; Khashabi et al., 2020). Our work is in line with Gururangan et al. (2020) where we investigate further the effectiveness of continued pre-training in the context of zero-shot cross-lingual SLU.

6.8 Conclusion

We systematically study the effectiveness of continued pre-training of a multilingual model on intermediate English unlabeled spoken language data for zero-shot cross-lingual tasks, namely intent classification and slot filling, on 8 languages. Our results show that the domain knowledge learned in English is transferable to other languages. Slot filling benefits more from continued pre-training than intent classification especially on Latin script languages. The gain from continued pre-training diminishes as we inject cross-lingual supervision through code-switching data augmentation in the fine-tuning stage. There are several factors that influence the effectiveness of the continued pre-training: (i) Using different language between pre-training and fine-tuning can hamper performance and introduce instability in the model training, which can be alleviated with code switching. (ii) Domain similarity is important. The more similar – in terms of data distribution – the intermediate data to the target dataset yields better performance. We believe that our findings could also be applicable to broader NLP tasks in the context of zero-shot cross-lingual settings.

Chapter 7

Conclusion & Future Work

7.1 Conclusion

This thesis focuses on investigating transfer learning and data augmentation methods for low-resource NLU in ToD systems. We surveyed recent developments on neural methods for task-oriented NLU and existing approaches when scaling models to new domains or languages with limited labeled data. We characterize existing low-resource NLU approaches in terms of their methods and auxiliary requirements and identify their challenges. As contributions, we conduct an investigation and propose solutions to address these challenges. In this chapter, we conclude the thesis and outline future research directions within task-oriented NLU.

In Chapter 4, we described our investigation on using non-conversational text as auxiliary data in the absence of a task-oriented dataset as a source of transfer. In particular, we use non-conversational text annotated with NER or Semantic Tagging as auxiliary tasks similar to slot filling tasks in NLU. We use multi-task learning (MTL) models to incorporate auxiliary tasks and show that MTL with non-conversational auxiliary tasks consistently outperforms single-task learning models. Furthermore, we investigate the potential of data selection on the auxiliary data before conducting MTL; however, the results are negative.

In Chapter 5, we proposed non-gradient based augmentation methods, Lightweight Augmentation. Lightweight Augmentation incorporate token and sentence level

augmentation, which consists of meaning preserving token substitution and syntactic manipulation through dependency parse tree information. We showed that LIGHTWEIGHT AUGMENTATION despite its simplicity, is competitive with deep-learning based augmentation and effective on low-resource scenarios on English and five non-English datasets. Combining transfer learning through BERT fine-tuning and LIGHTWEIGHT AUGMENTATION yields the best performance, which suggests that transfer learning and data augmentation are additive to performance.

In Chapter 6, we investigate the potential of performing continued pre-training of a pre-trained multilingual transformer model in the context of zero-shot cross-lingual NLU. In a zero-shot scenario where only an English dataset is available for training, continued pre-training is effective across languages. The effectiveness is especially evident for languages with Latin script systems. We also emphasize that using different languages between continued pre-training and fine-tuning can be detrimental to performance. The domain similarity between the intermediate data used in continued pre-training and the fine-tuning data is essential for performance benefit.

Overall, this thesis has investigated transfer learning and data augmentation approaches in different low-resource scenarios. In most cases, both transfer learning and data augmentation can improve model performance on NLU tasks. From the subtasks' perspective, observing the overall results, slot filling obtains the most benefit compared to intent classification in the majority of the cases. The margin of improvements can be affected by several factors, such as the size of the training instance in the target setting and characteristics of auxiliary data or pre-trained models. As the number of training instances in the target setting increases, the benefit of transfer learning and data augmentation becomes less evident. Especially for data augmentation, label preserving operations are a crucial factor. The characteristics of the auxiliary data, such as the domain similarity between the auxiliary data and the target data, is also an essential factor. When we use models based on large pre-trained language models (e.g., BERT, ALBERT, RoBERTa), while they still benefit from data augmentation, the improvement is less than smaller models such as LSTM based models.

7.2 Future Work

Based on the works that we have pursued in this thesis, there are some potential future directions for further research in the context of low-resource scenarios or even in a more general case of task-oriented NLU. First, the low-resource scenario that we consider is either data scarcity in a domain (Chapter 4 and 5) or language (Chapter 6). It could be the case that the data is scarce in *both* domain and language dimensions which require a transfer learning method that considers both dimensions simultaneously. There have been some recent works that pursue this specific low-resource scenario (Liu et al., 2021b; Razumovskaia et al., 2021). In addition to that, there are some works (Yu et al., 2021; Hou et al., 2021) which explore *few-shot scenarios* where only few examples are available in the target domain. Second, it would be interesting to leverage unlabeled data from live traffic. In real situations, personal digital assistants such as Google Home, Apple Siri, and Amazon Alexa, receive live traffic data from real users. This large amount of unlabeled data from live traffic is a potential data source for model training, in addition to in-house annotated data. Unlabeled live data are likely different from in-house data, as they can contain more diverse, noisy, and irrelevant utterances. In this situation, existing methods to tap on unlabeled data, such as semi-supervised learning, still face unique challenges to handle live data. It is worth noting that a bottleneck in this direction is that working on live data in academic settings is not trivial. Some works explore this line of research by applying semi-supervised learning (Cho et al., 2019) and also data selection (Do and Gaspers, 2019) mechanism. Third, regarding the evaluation, existing neural approaches are typically evaluated on *single-intent* utterance. However, in a real-world scenario, users may indicate *multiple-intent* in an utterance, e.g., "*Show me all flights from Atlanta to London and get the cost*" (Gangadharaiah and Narayanaswamy, 2019) or even expressing multiple sentences in one single turn. The evaluation of dialogue models on standard benchmarks often overestimates the model's performance in real-world settings. One of the main causes is that there is a gap between task-oriented datasets and real-world conversations. Typically, task-oriented datasets are constructed by crowd-workers following certain templates or instructions suggested by the Wizard of Oz methodology. As a result, utterances in the dataset tend to be semantically precise and clear, which resembles an ideal situation. On the other hand, in real-world conversations, there can be phenomena, such as speech errors and disfluencies, typos, out-of-domain

utterances, unseen entities, paraphrasing, and simplification, which are challenging for automatic systems. Therefore, it is important to evaluate the robustness of dialogue models against a range of phenomena they might encounter in real-world conversations, which go beyond the standard evaluation datasets. Recent initiatives introduce evaluation on the robustness of dialogue systems, addressing aspects not covered in current standard benchmarks. Peng et al. (2021) introduce RADDLE, an evaluation benchmark with a *robustness checklist* that covers phenomena such as paraphrase, verbosity, simplification, typos, etc. Their framework supports evaluation for Natural Language Understanding, Dialogue State Tracking, Dialogue Policy, and Natural Language Generation tasks. They evaluate the existing state-of-the-art models on these tasks and found that most models perform unsatisfactorily on these robustness evaluations, which suggests that there is still ample room for model improvements. Similarly, Liu et al. (2021a) propose a model-agnostic toolkit, LAUG, which generates natural language perturbations for evaluating the robustness issues in task-oriented dialog in terms of language variety, speech characteristics, and noise perturbation.

Bibliography

Abzianidze, L., Bjerva, J., Evang, K., Haagsma, H., van Noord, R., Ludmann, P., Nguyen, D.-D., and Bos, J. (2017). The Parallel Meaning Bank: Towards a Multilingual Corpus of Translations Annotated with Compositional Meaning Representations. In *EACL*.

Abzianidze, L. and Bos, J. (2017). Towards Universal Semantic Tagging. In *IWCS 2017 — 12th International Conference on Computational Semantics — Short papers*.

Adiwardana, D., Luong, M.-T., So, D. R., Hall, J., Fiedel, N., Thoppilan, R., Yang, Z., Kulshreshtha, A., Nemade, G., Lu, Y., and Le, Q. V. (2020). Towards a human-like open-domain chatbot. *ArXiv*, abs/2001.09977.

Alonso, H. M. and Plank, B. (2017). When is multitask learning effective? semantic sequence prediction under varying data conditions. In *EACL 2017-15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1–10.

Anaby-Tavor, A., Carmeli, B., Goldbraich, E., Kantor, A., Kour, G., Shlomov, S., Tepper, N., and Zwerdling, N. (2020). Do not have enough data? deep learning to the rescue! In *AAAI*.

Arase, Y. and Tsujii, J. (2019). Transfer fine-tuning: A BERT case study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5393–5404, Hong Kong, China. Association for Computational Linguistics.

Artetxe, M., Labaka, G., and Agirre, E. (2017). Learning bilingual word embeddings with (almost) no bilingual data. In *ACL*.

Asri, L. E., He, J., and Suleman, K. (2016). A sequence-to-sequence model for user simulation in spoken dialogue systems. In Morgan, N., editor, *Interspeech 2016, 17th*

Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016, pages 1151–1155. ISCA.

Austin, J. L. (1962). *How to do things with words*. William James Lectures. Oxford University Press.

Axelrod, A., He, X., and Gao, J. (2011). Domain adaptation via pseudo in-domain data selection. In *EMNLP*.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley Framenet Project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., and Schneider, N. (2013). Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186.

Bapna, A., Tür, G., Hakkani-Tür, D., and Heck, L. P. (2017). Towards zero-shot frame semantic parsing for domain scaling. In Lacerda, F., editor, *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, pages 2476–2480. ISCA.

Bellegarda, J. R. (2013). Large-scale personal assistant technology deployment: the siri experience. In Bimbot, F., Cerisara, C., Fougeron, C., Gravier, G., Lamel, L., Pellegrino, F., and Perrier, P., editors, *INTERSPEECH*, pages 2029–2033. ISCA.

Bellomaria, V., Castellucci, G., Favalli, A., and Romagnoli, R. (2019). Almawave-slu: A new dataset for SLU in italian. In Bernardi, R., Navigli, R., and Semeraro, G., editors, *Proceedings of the Sixth Italian Conference on Computational Linguistics, Bari, Italy, November 13-15, 2019*, volume 2481 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Beltagy, I., Lo, K., and Cohan, A. (2019). SciBERT: A pretrained language model for scientific text. In *EMNLP/IJCNLP*.

Bingel, J. and Søgaard, A. (2017). Identifying beneficial task relations for multi-task learning in deep neural networks. *EACL 2017*, page 164.

Bjerva, J., Plank, B., and Bos, J. (2016). Semantic tagging with deep residual networks. In *COLING*.

Blitzer, J., Dredze, M., and Pereira, F. C. (2007). Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*.

Bobrow, D., Kaplan, R., Kay, M., Norman, D., Thompson, H. S., and Winograd, T. (1977). Gus, a frame-driven dialog system. *Artif. Intell.*, 8:155–173.

Bohus, D. and Rudnicky, A. I. (2009). The ravenclaw dialog management framework: Architecture and systems. *Comput. Speech Lang.*, 23:332–361.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Bonneau-Maynard, H., Rosset, S., Ayache, C., Kuhn, A., and Mostefa, D. (2005). Semantic annotation of the french media dialog corpus. In *INTERSPEECH 2005 - Eurospeech, 9th European Conference on Speech Communication and Technology, Lisbon, Portugal, September 4-8, 2005*, pages 3457–3460. ISCA.

Brochu, E., Cora, V. M., and De Freitas, N. (2010). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*.

Budzianowski, P., Wen, T.-H., Tseng, B.-H., Casanueva, I., Ultes, S., Ramadan, O., and Gašić, M. (2018). MultiWOZ - a large-scale multi-domain wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Callison-Burch, C., Koehn, P., and Osborne, M. (2006). Improved statistical machine translation using paraphrases. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 17–24, New York City, USA. Association for Computational Linguistics.

Cao, Y., Lu, K., Chen, X., and Zhang, S. (2020). Adaptive dialog policy learning with hindsight and user modeling. In *SIGDIAL*.

Caruana, R. (1997). Multitask learning. *Machine learning*, 28(1):41–75.

Changpinyo, S., Hu, H., and Sha, F. (2018). Multi-task learning for sequence tagging: An empirical study. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2965–2977, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Chen, Q., Zhuo, Z., and Wang, W. (2019). Bert for joint intent classification and slot filling. *ArXiv*, abs/1902.10909.

Chen, Y., Hakanni-Tur, D., Tür, G., Çelikyilmaz, A., Gao, J., and Deng, L. (2016). Syntax or Semantics? Knowledge-guided Joint Semantic Frame Parsing. In *2016 IEEE Spoken Language Technology Workshop, SLT 2016, San Diego, CA, USA, December 13-16, 2016*, pages 348–355.

Chiu, J. P. and Nichols, E. (2016). Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Cho, E., Xie, H., Lalor, J. P., Kumar, V., and Campbell, W. M. (2019). Efficient semi-supervised learning for natural language understanding by optimizing diversity. In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019, Singapore, December 14-18, 2019*, pages 1077–1084. IEEE.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Moschitti, A., Pang, B., and Daelemans, W., editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL.

Christodoulopoulos, C. and Steedman, M. (2015). A massively parallel corpus: the bible in 100 languages. In *Language Resources and Evaluation*.

Clark, H. and Schaefer, E. (1987). Collaborating on contributions to conversations. *Language Cognition and Neuroscience*, pages 19–41.

Clark, H. H. and Brennan, S. E. (1991). Grounding in communication. In Resnick, L., Levine, J., and Teasley, S., editors, *Perspectives on Socially Shared Cognition*, pages 127–149. American Psychological Association.

Colby, K., Weber, S., and Hilf, F. D. (1971). Artificial paranoia. *Artif. Intell.*, 2:1–25.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. P. (2011). Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. In *ACL*.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., and Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*.

Coucke, A., Saade, A., Ball, A., Bluche, T., Caulier, A., Leroy, D., Doumouro, C., Gisselbrecht, T., Caltagirone, F., Lavril, T., Primet, M., and Dureau, J. (2018). Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *ArXiv*, abs/1805.10190.

Dai, X. and Adel, H. (2020). An analysis of simple data augmentation for named entity recognition. In Scott, D., Bel, N., and Zong, C., editors, *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 3861–3867. International Committee on Computational Linguistics.

Dai, X., Karimi, S., Hachey, B., and Paris, C. (2020). Cost-effective selection of pretraining data: A case study of pretraining bert on social media. In *EMNLP*.

Dathathri, S., Madotto, A., Lan, J., Hung, J., Frank, E., Molino, P., Yosinski, J., and Liu, R. (2020). Plug and play language models: A simple approach to controlled text generation. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Daumé III, H. (2007). Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Do, Q. N. T. and Gaspers, J. (2019). Cross-lingual transfer learning with data selection for large-scale spoken language understanding. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1455–1460. Association for Computational Linguistics.

Donahue, C., Lee, M., and Liang, P. (2020). Enabling language models to fill in the blanks. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. R., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2492–2501. Association for Computational Linguistics.

Duchi, J. C., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159.

E, H., Niu, P., Chen, Z., and Song, M. (2019). A novel bi-directional interrelated model for joint intent detection and slot filling. In Korhonen, A., Traum, D. R., and Màrquez, L., editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5467–5471. Association for Computational Linguistics.

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211.

Fadaee, M., Bisazza, A., and Monz, C. (2017). Data augmentation for low-resource neural machine translation. In Barzilay, R. and Kan, M., editors, *Proceedings of the*

55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers, pages 567–573. Association for Computational Linguistics.

Fan, A., Lewis, M., and Dauphin, Y. N. (2018). Hierarchical neural story generation. In Gurevych, I. and Miyao, Y., editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 889–898. Association for Computational Linguistics.

Feng, S. Y., Gangal, V., Wei, J., Chandar, S., Vosoughi, S., Mitamura, T., and Hovy, E. (2021). A survey of data augmentation approaches for nlp. *Findings of ACL*.

Gangadharaiah, R. and Narayanaswamy, B. (2019). Joint multiple intent detection and slot labeling for goal-oriented dialog. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 564–569. Association for Computational Linguistics.

Gao, J., Galley, M., and Li, L. (2019). Neural approaches to conversational AI. *Found. Trends Inf. Retr.*, 13(2-3):127–298.

Garg, S., Vu, T., and Moschitti, A. (2020). Tanda: Transfer and adapt pre-trained transformer models for answer sentence selection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7780–7788.

Gehman, S., Gururangan, S., Sap, M., Choi, Y., and Smith, N. A. (2020). RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Giampiccolo, D., Magnini, B., Dagan, I., and Dolan, B. (2007). The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague. Association for Computational Linguistics.

Gong, Y., Luo, X., Zhu, Y., Ou, W., Li, Z., Zhu, M., Zhu, K. Q., Duan, L., and Chen, X. (2019). Deep cascade multi-task learning for slot filling in online shopping assistant. In *AAAI 2019*.

Goo, C., Gao, G., Hsu, Y., Huo, C., Chen, T., Hsu, K., and Chen, Y. (2018). 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, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 753–757.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Goyal, A. K., Metallinou, A., and Matsoukas, S. (2018). Fast and scalable expansion of natural language understanding functionality for intelligent agents. In Bangalore, S., Chu-Carroll, J., and Li, Y., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 3 (Industry Papers)*, pages 145–152. Association for Computational Linguistics.

Grishman, R. and Sundheim, B. (1996). Message understanding conference- 6: A brief history. In *16th International Conference on Computational Linguistics, Proceedings of the Conference, COLING 1996, Center for Sprogteknologi, Copenhagen, Denmark, August 5-9, 1996*, pages 466–471.

Grosz, B. J., Appelt, D. E., Martin, P. A., and Pereira, F. C. (1987). Team: An experiment in the design of transportable natural-language interfaces. *Artificial Intelligence*, 32(2):173–243.

Guerini, M., Magnolini, S., Balaraman, V., and Magnini, B. (2018). Toward zero-shot entity recognition in task-oriented conversational agents. In Komatani, K., Litman, D. J., Yu, K., Cavedon, L., Nakano, M., and Papangelis, A., editors, *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue, Melbourne, Australia, July 12-14, 2018*, pages 317–326. Association for Computational Linguistics.

Gulordava, K., Bojanowski, P., Grave, E., Linzen, T., and Baroni, M. (2018). Colorless green recurrent networks dream hierarchically. In Walker, M. A., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 1195–1205. Association for Computational Linguistics.

Guo, D., Tur, G., Yih, W.-t., and Zweig, G. (2014). Joint semantic utterance classification and slot filling with recursive neural networks. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pages 554–559. IEEE.

Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., and Smith, N. A. (2020). Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Hakkani-Tür, D. Z., Tür, G., Çelikyilmaz, A., Chen, Y.-N., Gao, J., Deng, L., and Wang, Y.-Y. (2016). Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. In *INTERSPEECH*.

Han, X. and Eisenstein, J. (2019). Unsupervised domain adaptation of contextualized embeddings for sequence labeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4238–4248, Hong Kong, China. Association for Computational Linguistics.

hao Su, P., Mrksic, N., Casanueva, I., and Vulic, I. (2018). Deep learning for conversational ai. In *NAACL*.

He, T., Liu, J., Cho, K., Ott, M., Liu, B., Glass, J., and Peng, F. (2019). Analyzing the forgetting problem in the pretrain-finetuning of dialogue response models. *arXiv: Computation and Language*.

He, Y. and Young, S. J. (2003). A data-driven spoken language understanding system. *2003 IEEE Workshop on Automatic Speech Recognition and Understanding (IEEE Cat. No.03EX721)*, pages 583–588.

Hedderich, M. A., Lange, L., Adel, H., Strötgen, J., and Klakow, D. (2021). A survey on recent approaches for natural language processing in low-resource scenarios. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2545–2568, Online. Association for Computational Linguistics.

Hemphill, C. T., Godfrey, J. J., and Doddington, G. R. (1990). The ATIS spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, USA, June 24-27, 1990*. Morgan Kaufmann.

Henderson, M., Thomson, B., and Young, S. (2014). Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 292–299.

Henderson, M. and Vulić, I. (2021). ConVEx: Data-efficient and few-shot slot labeling. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3375–3389, Online. Association for Computational Linguistics.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9:1735–1780.

Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. (2020). The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Hou, Y., Che, W., Lai, Y., Zhou, Z., Liu, Y., Liu, H., and Liu, T. (2020). Few-shot slot tagging with collapsed dependency transfer and label-enhanced task-adaptive projection network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1381–1393, Online. Association for Computational Linguistics.

Hou, Y., Lai, Y., Chen, C., Che, W., and Liu, T. (2021). Learning to bridge metric spaces: Few-shot joint learning of intent detection and slot filling. In Zong, C., Xia, F., Li, W., and Navigli, R., editors, *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pages 3190–3200. Association for Computational Linguistics.

Hou, Y., Liu, Y., Che, W., and Liu, T. (2018a). Sequence-to-sequence data augmentation for dialogue language understanding. In Bender, E. M., Derczynski, L., and Isabelle, P., editors, *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pages 1234–1245. Association for Computational Linguistics.

Hou, Y., Liu, Y., Che, W., and Liu, T. (2018b). Sequence-to-sequence data augmentation for dialogue language understanding. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1234–1245, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Huang, M., Zhu, X., and Gao, J. (2020). Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38:1 – 32.

Jaech, A., Heck, L. P., and Ostendorf, M. (2016). Domain Adaptation of Recurrent Neural Networks for Natural Language Understanding. In *INTERSPEECH*.

Jeong, M. and Lee, G. G. (2008). Triangular-chain conditional random fields. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(7):1287–1302.

Jha, R., Marin, A., Shivaprasad, S., and Zitouni, I. (2018). Bag of experts architectures for model reuse in conversational language understanding. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 153–161. Association for Computational Linguistics.

Jordan, M. I. (1997). Serial order: A parallel distributed processing approach. In *Advances in psychology*, volume 121, pages 471–495. Elsevier.

Joulin, A., Bojanowski, P., Mikolov, T., Jégou, H., and Grave, E. (2018). Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *EMNLP*.

Jurafsky, D. and Martin, J. H. (2009). Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd edition. In *Prentice Hall series in artificial intelligence*.

Kale, M. and Rastogi, A. (2020). Template guided text generation for task oriented dialogue. In *EMNLP*.

Kamholz, D., Pool, J., and Colowick, S. (2014). Panlex: Building a resource for panlingual lexical translation. In *LREC*.

Kennard, N. N., Hakkani-Tür, D. Z., Walker, M., and Heck, L. (2017). To plan or not to plan? discourse planning in slot-value informed sequence to sequence models for language generation. In *INTERSPEECH*.

Keung, P., Lu, Y., Salazar, J., and Bhardwaj, V. (2020). Don't Use English Dev: On the Zero-Shot Cross-Lingual Evaluation of Contextual Embeddings. In *EMNLP*.

Khashabi, D., Min, S., Khot, T., Sabharwal, A., Tafjord, O., Clark, P., and Hajishirzi, H. (2020). UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Kim, S. and Banchs, R. E. (2014). Sequential labeling for tracking dynamic dialog states. In *SIGDIAL Conference*.

Kim, Y., Stratos, K., and Sarikaya, R. (2016). Domainless Adaptation by Constrained Decoding on a Schema Lattice. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2051–2060.

Kim, Y.-B., Stratos, K., and Kim, D. (2017). Domain attention with an ensemble of experts. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653. Association for Computational Linguistics.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P. and Ba, J. (2015). Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Kobayashi, S. (2018). Contextual augmentation: Data augmentation by words with paradigmatic relations. In Walker, M. A., Ji, H., and Stent, A., editors, *Proceedings*

of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers), pages 452–457. Association for Computational Linguistics.

Krishnan, J., Anastasopoulos, A., Purohit, H., and Rangwala, H. (2021). Multilingual code-switching for zero-shot cross-lingual intent prediction and slot filling. *ArXiv*, abs/2103.07792.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84 – 90.

Kulshreshtha, S., Redondo Garcia, J. L., and Chang, C.-Y. (2020). Cross-lingual alignment methods for multilingual BERT: A comparative study. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 933–942, Online. Association for Computational Linguistics.

Kumar, V., Choudhary, A., and Cho, E. (2020). Data augmentation using pre-trained transformer models. *ArXiv*, abs/2003.02245.

Kurata, G., Xiang, B., and Zhou, B. (2016a). Labeled data generation with encoder-decoder lstm for semantic slot filling. In *INTERSPEECH*.

Kurata, G., Xiang, B., Zhou, B., and Yu, M. (2016b). Leveraging sentence-level information with encoder LSTM for semantic slot filling. In Su, J., Carreras, X., and Duh, K., editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2077–2083. The Association for Computational Linguistics.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics.

Lample, G. and Conneau, A. (2019). Cross-lingual language model pretraining. In *NeurIPS*.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2020). ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Lauscher, A., Ravishankar, V., Vulić, I., and Glavaš, G. (2020). From zero to hero: On the limitations of zero-shot language transfer with multilingual Transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499, Online. Association for Computational Linguistics.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, H., Lee, J., and Kim, T.-Y. (2019). Sumbt: Slot-utterance matching for universal and scalable belief tracking. In *ACL*.

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. (2020). BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36:1234 – 1240.

Lee, S. and Jha, R. (2019). Zero-shot adaptive transfer for conversational language understanding. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6642–6649. AAAI Press.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Li, C., Li, L., and Qi, J. (2018). A self-attentive model with gate mechanism for spoken language understanding. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii,

J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 3824–3833. Association for Computational Linguistics.

Li, H., Arora, A., Chen, S., Gupta, A., Gupta, S., and Mehdad, Y. (2021a). MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962, Online. Association for Computational Linguistics.

Li, J., Galley, M., Brockett, C., Gao, J., and Dolan, B. (2016). A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Li, J., Tang, T., Zhao, W. X., and Wen, J. (2021b). Pretrained language models for text generation: A survey. *CoRR*, abs/2105.10311.

Li, Y., Cohn, T., and Baldwin, T. (2017). Robust training under linguistic adversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 21–27, Valencia, Spain. Association for Computational Linguistics.

Lin, H., Xiang, L., Zhou, Y., Zhang, J., and Zong, C. (2021). Augmenting slot values and contexts for spoken language understanding with pretrained models. *Interspeech*, abs/2108.08451.

Lin, J. (1991). Divergence measures based on the shannon entropy. *IEEE Trans. Information Theory*, 37:145–151.

Lison, P. and Tiedemann, J. (2016). Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *LREC*.

Liu, B. and Lane, I. (2015). Recurrent neural network structured output prediction for spoken language understanding.

Liu, B. and Lane, I. (2016a). Attention-based recurrent neural network models for joint intent detection and slot filling. In Morgan, N., editor, *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, pages 685–689. ISCA.

Liu, B. and Lane, I. (2016b). Joint Online Spoken Language Understanding and Language Modeling With Recurrent Neural Networks. In *Proceedings of the SIGDIAL 2016 Conference, The 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 13-15 September 2016, Los Angeles, CA, USA*, pages 22–30.

Liu, B. and Lane, I. (2017). Multi-domain adversarial learning for slot filling in spoken language understanding. In *NIPS Workshop on Conversational AI*.

Liu, J., Pasupat, P., Cyphers, S., and Glass, J. R. (2013). Asgard: A portable architecture for multilingual dialogue systems. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 8386–8390. IEEE.

Liu, J., Takanobu, R., Wen, J., Wan, D., Li, H., Nie, W., Li, C., Peng, W., and Huang, M. (2021a). Robustness testing of language understanding in task-oriented dialog. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2467–2480, Online. Association for Computational Linguistics.

Liu, M., Song, Y., Zou, H., and Zhang, T. (2019a). Reinforced training data selection for domain adaptation. In *ACL*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019b). RoBERTa: A robustly optimized BERT pretraining approach. *ArXiv*, abs/1907.11692.

Liu, Z., Shin, J., Xu, Y., Winata, G. I., Xu, P., Madotto, A., and Fung, P. (2019c). Zero-shot cross-lingual dialogue systems with transferable latent variables. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1297–1303. Association for Computational Linguistics.

Liu, Z., Winata, G. I., Lin, Z., Xu, P., and Fung, P. (2020a). Attention-informed mixed-language training for zero-shot cross-lingual task-oriented dialogue systems. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8433–8440. AAAI Press.

Liu, Z., Winata, G. I., Xu, P., and Fung, P. (2020b). Coach: A coarse-to-fine approach for cross-domain slot filling. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. R., editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 19–25. Association for Computational Linguistics.

Liu, Z., Winata, G. I., Xu, P., and Fung, P. (2021b). X2Parser: Cross-lingual and cross-domain framework for task-oriented compositional semantic parsing. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 112–127, Online. Association for Computational Linguistics.

Liu, Z., Winata, G. I., Xu, P., Lin, Z., and Fung, P. (2020c). Cross-lingual spoken language understanding with regularized representation alignment. In Webber, B., Cohn, T., He, Y., and Liu, Y., editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7241–7251. Association for Computational Linguistics.

Logeswaran, L., Chang, M.-W., Lee, K., Toutanova, K., Devlin, J., and Lee, H. (2019). Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460, Florence, Italy. Association for Computational Linguistics.

Louvan, S. and Magnini, B. (2018a). Exploring named entity recognition as an auxiliary task for slot filling in conversational language understanding. In *Proceedings of the 2018 EMNLP Workshop SCAI: The 2nd International Workshop on Search-Oriented Conversational AI*, pages 74–80, Brussels, Belgium. Association for Computational Linguistics.

Louvan, S. and Magnini, B. (2018b). From General to Specific : Leveraging Named Entity Recognition for Slot Filling in Conversational Language Understanding. In *CLiC-it*.

Louvan, S. and Magnini, B. (2019). Leveraging non-conversational tasks for low resource slot filling: Does it help? In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 85–91, Stockholm, Sweden. Association for Computational Linguistics.

Louvan, S. and Magnini, B. (2020a). How far can we go with data selection? a case study on semantic sequence tagging tasks. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pages 15–21, Online. Association for Computational Linguistics.

Louvan, S. and Magnini, B. (2020b). Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 480–496, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Louvan, S. and Magnini, B. (2020c). Simple data augmentation for multilingual nlu in task oriented dialogue systems. In *CLiC-it*.

Louvan, S. and Magnini, B. (2020d). Simple is better! lightweight data augmentation for low resource slot filling and intent classification. In *Proceedings of the 34th Pacific Asia Conference on Language, Information and Computation*, pages 167–177, Hanoi, Vietnam. Association for Computational Linguistics.

Luong, M.-T., Le, Q. V., Sutskever, I., Vinyals, O., and Kaiser, L. (2016). Multi-task sequence to sequence learning. *ICLR*, abs/1511.06114.

Ma, X. and Hovy, E. (2016). End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1064–1074.

Madotto, A., Wu, C.-S., and Fung, P. (2018). Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *ACL*.

Magnini, B. and Louvan, S. (2021). *Understanding Dialogue for Human Communication*, pages 1–43. Springer International Publishing, Cham.

Majumder, N., Hong, P., Peng, S., Lu, J., Ghosal, D., Gelbukh, A., Mihalcea, R., and Poria, S. (2020). Mime: Mimicking emotions for empathetic response generation. *ArXiv*, abs/2010.01454.

Manning, C. D. (2015). Computational linguistics and deep learning. *Computational Linguistics*, 41:701–707.

Marasovic, A. and Frank, A. (2018). SRL4ORL: Improving Opinion Role Labelling using Multi-task Learning with Semantic Role Labeling. In *NAACL-HLT*.

McTear, M. (2020). *Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots*. Morgan and Claypool Publishers.

Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tür, D. Z., He, X., Heck, L. P., Tür, G., Yu, D., and Zweig, G. (2015). Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23:530–539.

Mesnil, G., He, X., Deng, L., and Bengio, Y. (2013). Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *INTERSPEECH*.

Metallinou, A., Bohus, D., and Williams, J. (2013). Discriminative state tracking for spoken dialog systems. In *ACL*.

Mikolov, T., Chen, K., Corrado, G. S., and Dean, J. (2013). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Mikolov, T., Kombrink, S., Deoras, A., Burget, L., and Cernocky, J. (2011). Rnnlm - recurrent neural network language modeling toolkit.

Moschitti, A., Riccardi, G., and Raymond, C. (2007). Spoken language understanding with kernels for syntactic/semantic structures. In *2007 IEEE Workshop on Automatic Speech Recognition & Understanding (ASRU)*, pages 183–188. IEEE.

Mou, L., Meng, Z., Yan, R., Li, G., Xu, Y., Zhang, L., and Jin, Z. (2016). How Transferable are Neural Networks in NLP Applications? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 479–489. Association for Computational Linguistics.

Mrksic, N., Séaghdha, D. Ó., Thomson, B., Gasic, M., Su, P., Vandyke, D., Wen, T., and Young, S. J. (2015). Multi-domain dialog state tracking using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 794–799. The Association for Computer Linguistics.

Mrksic, N., Séaghdha, D. Ó., Wen, T.-H., Thomson, B., and Young, S. (2017). Neural belief tracker: Data-driven dialogue state tracking. In *ACL*.

Murthy, V. R., Kunchukuttan, A., and Bhattacharyya, P. (2018). Judicious selection of training data in assisting language for multilingual neural ner. In *ACL*.

Palmer, M., Gildea, D., and Kingsbury, P. (2005). The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational linguistics*, 31(1):71–106.

Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22:1345–1359.

Peng, B., Li, C., Zhang, Z., Zhu, C., Li, J., and Gao, J. (2021). RADDLE: An evaluation benchmark and analysis platform for robust task-oriented dialog systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4418–4429, Online. Association for Computational Linguistics.

Peng, B., Zhu, C., Zeng, M., and Gao, J. (2020). Data augmentation for spoken language understanding via pretrained models. *CoRR*, abs/2004.13952.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *EMNLP*.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep Contextualized Word Representations". In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Pfeiffer, J., Vulić, I., Gurevych, I., and Ruder, S. (2020). MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Pires, T., Schlinger, E., and Garrette, D. (2019). How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy. Association for Computational Linguistics.

Plank, B. (2011). Domain adaptation for parsing.

Plank, B. (2016). What to do about non-standard (or non-canonical) language in NLP. *arXiv preprint arXiv:1608.07836*.

Plank, B., Søgaard, A., and Goldberg, Y. (2016). Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 412–418.

Pradhan, S., Moschitti, A., Xue, N., Ng, H. T., Björkelund, A., Uryupina, O., Zhang, Y., and Zhong, Z. (2013). Towards Robust Linguistic Analysis using OntoNotes. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 143–152.

Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., and Zhang, Y. (2012). CoNLL-2012 Shared Task: Modeling Multilingual Unrestricted Coreference in OntoNotes. In *EMNLP-CoNLL Shared Task*.

Price, P. J. (1990). Evaluation of Spoken Language Systems: The ATIS Domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Qin, L., Che, W., Li, Y., Wen, H., and Liu, T. (2019). A stack-propagation framework with token-level intent detection for spoken language understanding. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2078–2087. Association for Computational Linguistics.

Qin, L., Ni, M., Zhang, Y., and Che, W. (2020a). Cosda-ml: Multi-lingual code-switching data augmentation for zero-shot cross-lingual NLP. In Bessiere, C., editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 3853–3860. ijcai.org.

Qin, L., Xie, T., Che, W., and Liu, T. (2021). A survey on spoken language understanding: Recent advances and new frontiers.

Qin, L., Xu, X., Che, W., Zhang, Y., and Liu, T. (2020b). Dynamic fusion network for multi-domain end-to-end task-oriented dialog. In *ACL*.

Quteineh, H., Samothrakis, S., and Sutcliffe, R. (2020). Textual data augmentation for efficient active learning on tiny datasets. In *EMNLP*.

Radford, A. (2018). Improving language understanding by generative pre-training.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners.

Ramponi, A. and Plank, B. (2020). Neural unsupervised domain adaptation in NLP—A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6838–6855, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Ramshaw, L. and Marcus, M. (1995a). Text Chunking using Transformation-Based Learning. In *Third Workshop on Very Large Corpora*.

Ramshaw, L. and Marcus, M. (1995b). Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Rashkin, H., Smith, E. M., Li, M., and Boureau, Y.-L. (2018). I know the feeling: Learning to converse with empathy. *ArXiv*, abs/1811.00207.

Raux, A., Langner, B., Bohus, D., Black, A., and Eskénazi, M. (2005). Let's go public! taking a spoken dialog system to the real world. In *INTERSPEECH*.

Ravuri, S. V. and Stolcke, A. (2015). Recurrent neural network and LSTM models for lexical utterance classification. In *INTERSPEECH 2015, 16th Annual Conference of*

the International Speech Communication Association, Dresden, Germany, September 6-10, 2015, pages 135–139. ISCA.

Raymond, C. and Riccardi, G. (2007). Generative and discriminative algorithms for spoken language understanding. In *INTERSPEECH*.

Razumovskaia, E., Glavavs, G., Majewska, O., Ponti, E., Korhonen, A., and Vuli'c, I. (2021). Crossing the conversational chasm: A primer on natural language processing for multilingual task-oriented dialogue systems.

Reimers, N. and Gurevych, I. (2017). Reporting Score Distributions Makes a Difference: Performance Study of LSTM-networks for Sequence Tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 338–348, Copenhagen, Denmark.

Reimers, N. and Gurevych, I. (2018). Why Comparing Single Performance Scores Does Not Allow to Draw Conclusions About Machine Learning Approaches. *CoRR*, abs/1803.09578.

Ritter, A., Cherry, C., and Dolan, W. (2011). Data-driven response generation in social media. In *EMNLP*.

Rojas-Barahona, L., Gaić, M., Mrksic, N., hao Su, P., Ultes, S., Wen, T.-H., Young, S., and Vandyke, D. (2017). A network-based end-to-end trainable task-oriented dialogue system. In *EACL*.

Rosenblatt, F. (1957). The perceptron - a perceiving and recognizing automaton. Technical Report 85-460-1, Cornell Aeronautical Laboratory, Ithaca, New York.

Rosenstein, M. (2005). To transfer or not to transfer. In *NIPS 2005*.

Ruder, S. (2019). *Neural Transfer Learning for Natural Language Processing*. PhD thesis, National University of Ireland, Galway.

Ruder, S. and Plank, B. (2017a). Learning to select data for transfer learning with bayesian optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 372–382, Copenhagen, Denmark. Association for Computational Linguistics.

Ruder, S. and Plank, B. (2017b). Learning to select data for transfer learning with Bayesian optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 372–382, Copenhagen, Denmark. Association for Computational Linguistics.

Rumelhart, D., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323:533–536.

Sahin, G. G. and Steedman, M. (2018). Data augmentation via dependency tree morphing for low-resource languages. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J., editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 5004–5009. Association for Computational Linguistics.

Sang, E. F. T. K. and Meulder, F. D. (2003). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In Daelemans, W. and Osborne, M., editors, *Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003*, pages 142–147. ACL.

Sanh, V., Wolf, T., and Ruder, S. (2019). A Hierarchical Multi-task Approach for Learning Embeddings from Semantic Tasks. *AAAI*.

Schröder, F. and Biemann, C. (2020). Estimating the influence of auxiliary tasks for multi-task learning of sequence tagging tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2971–2985, Online. Association for Computational Linguistics.

Schuster, M. and Paliwal, K. (1997). Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.*, 45:2673–2681.

Schuster, S., Gupta, S., Shah, R., and Lewis, M. (2019a). Cross-lingual transfer learning for multilingual task oriented dialog. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3795–3805. Association for Computational Linguistics.

Schuster, S., Gupta, S., Shah, R., and Lewis, M. (2019b). Cross-lingual transfer learning for multilingual task oriented dialog. 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)*, pages 3795–3805, Minneapolis, Minnesota. Association for Computational Linguistics.

Searle, J. R. (1969). *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press.

Sennrich, R., Haddow, B., and Birch, A. (2016). Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Shah, D., Gupta, R., Fayazi, A., and Hakkani-Tur, D. (2019). Robust zero-shot cross-domain slot filling with example values. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5484–5490, Florence, Italy. Association for Computational Linguistics.

Shan, Y., Li, Z., Zhang, J., Meng, F., Feng, Y., Niu, C., and Zhou, J. (2020). A contextual hierarchical attention network with adaptive objective for dialogue state tracking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6322–6333, Online. Association for Computational Linguistics.

Shang, L., Lu, Z., and Li, H. (2015). Neural responding machine for short-text conversation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1577–1586, Beijing, China. Association for Computational Linguistics.

Shum, H., He, X., and Li, D. (2018). From eliza to xiaoice: challenges and opportunities with social chatbots. *Frontiers of Information Technology & Electronic Engineering*, 19:10–26.

Siddhant, A., Goyal, A. K., and Metallinou, A. (2019). Unsupervised transfer learning for spoken language understanding in intelligent agents. In *The Thirty-Third AAAI*

Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019, pages 4959–4966. AAAI Press.

Smith, S. L., Turban, D. H. P., Hamblin, S., and Hammerla, N. Y. (2017). Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Søgaard, A. and Goldberg, Y. (2016a). Deep Multi-task Learning with Low Level Tasks Supervised at Lower Layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 231–235.

Søgaard, A. and Goldberg, Y. (2016b). Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235, Berlin, Germany. Association for Computational Linguistics.

Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., Nie, J.-Y., Gao, J., and Dolan, W. (2015). A neural network approach to context-sensitive generation of conversational responses. In *HLT-NAACL*.

Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Van Ess-Dykema, C., and Meteer, M. (2000). Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics*, 26(3):339–374.

Su, P.-H., Gašić, M., Mrkšić, N., Rojas-Barahona, L. M., Ultes, S., Vandyke, D., Wen, T.-H., and Young, S. (2016). On-line active reward learning for policy optimisation in spoken dialogue systems. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2431–2441, Berlin, Germany. Association for Computational Linguistics.

Su, S.-Y., Li, X., Gao, J., Liu, J., and Chen, Y.-N. (2018). Discriminative deep Dyna-Q: Robust planning for dialogue policy learning. In *Proceedings of the 2018 Conference*

on Empirical Methods in Natural Language Processing, pages 3813–3823, Brussels, Belgium. Association for Computational Linguistics.

Summers, C. and Dinneen, M. J. (2019). Improved mixed-example data augmentation. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1262–1270. IEEE.

Susanto, R. H. and Lu, W. (2017). Neural architectures for multilingual semantic parsing. In Barzilay, R. and Kan, M., editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers*, pages 38–44. Association for Computational Linguistics.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *NIPS*.

Tiedemann, J. (2012). Parallel data, tools and interfaces in opus. In *LREC*.

Tjong Kim Sang, E. F. and Buchholz, S. (2000). Introduction to the CoNLL-2000 shared task chunking. In *Fourth Conference on Computational Natural Language Learning and the Second Learning Language in Logic Workshop*.

Tjong Kim Sang, E. F. and De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Traum, D. and Nakatani, C. (2002). A two-level approach to coding dialogue for discourse structure: Activities of the 1998 dri working group on higher-level structures.

Traum, D. R. and Heeman, P. A. (1996). Utterance units in spoken dialogue. In Maier, E., Mast, M., and LuperFoy, S., editors, *Dialogue Processing in Spoken Language Systems, ECAI'96 Workshop, Budapest, Hungary, August 13, 1996, Revised Papers*, volume 1236 of *Lecture Notes in Computer Science*, pages 125–140. Springer.

Tuan, Y.-L., Chen, Y.-N., and Lee, H.-y. (2019). DyKgChat: Benchmarking dialogue generation grounding on dynamic knowledge graphs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th Interna-*

tional Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 1855–1865, Hong Kong, China. Association for Computational Linguistics.

Tur, G. and De Mori, R. (2011). *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Turing, A. (1950). Computing machinery and intelligence. In *The Philosophy of Artificial Intelligence*.

Upadhyay, S., Faruqui, M., Tür, G., Hakkani-Tür, D. Z., and Heck, L. (2018). (almost) zero-shot cross-lingual spoken language understanding. *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6034–6038.

van der Goot, R., Sharaf, I., Imankulova, A., Üstün, A., Stepanović, M., Ramponi, A., Khairunnisa, S. O., Komachi, M., and Plank, B. (2021). From masked language modeling to translation: Non-English auxiliary tasks improve zero-shot spoken language understanding. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2479–2497, Online. Association for Computational Linguistics.

van der Wees, M., Bisazza, A., Weerkamp, W., and Monz, C. (2015). What's in a domain? analyzing genre and topic differences in statistical machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 560–566, Beijing, China. Association for Computational Linguistics.

Vania, C., Kementchedjhieva, Y., Søgaard, A., and Lopez, A. (2019a). A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 1105–1116. Association for Computational Linguistics.

Vania, C., Kementchedjhieva, Y., Søgaard, A., and Lopez, A. (2019b). A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. In *EMNLP/IJCNLP*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5998–6008.

Vijayakumar, A. K., Cogswell, M., Selvaraju, R. R., Sun, Q., Lee, S., Crandall, D. J., and Batra, D. (2018). Diverse beam search for improved description of complex scenes. In *AAAI*.

Vinyals, O. and Le, Q. V. (2015). A neural conversational model. *ArXiv*, abs/1506.05869.

Vu, N. T., Gupta, P., Adel, H., and Schütze, H. (2016). Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6060–6064. IEEE.

Walker, M. and Whittaker, S. (1990). Mixed initiative in dialogue: An investigation into discourse segmentation. In *ACL*.

Wallace, R. (2009). The anatomy of a.l.i.c.e.

Wang, W. Y. and Yang, D. (2015). That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., and Marton, Y., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 2557–2563. The Association for Computational Linguistics.

Wang, Y., Shen, Y., and Jin, H. (2018a). A bi-model based rnn semantic frame parsing model for intent detection and slot filling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 309–314. Association for Computational Linguistics.

Wang, Y., Shen, Y., and Jin, H. (2018b). A bi-model based RNN semantic frame parsing model for intent detection and slot filling. In Walker, M. A., Ji, H., and Stent, A., editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 309–314. Association for Computational Linguistics.

Wang, Y.-Y., Deng, L., and Acero, A. (2005). Spoken language understanding. *IEEE Signal Processing Magazine*, 22(5):16–31.

Wei, J. and Zou, K. (2019a). EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6382–6388, Hong Kong, China. Association for Computational Linguistics.

Wei, J. W. and Zou, K. (2019b). EDA: easy data augmentation techniques for boosting performance on text classification tasks. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6381–6387. Association for Computational Linguistics.

Weizenbaum, J. (1966). Eliza—a computer program for the study of natural language communication between man and machine. *Commun. ACM*, 9:36–45.

Wen, T., Gasic, M., Mrksic, N., Su, P., Vandyke, D., and Young, S. J. (2015a). Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., and Marton, Y., editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1711–1721. The Association for Computational Linguistics.

Wen, T.-H., Gašić, M., Kim, D., Mrkšić, N., Su, P.-H., Vandyke, D., and Young, S. (2015b). Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. In *Proceedings of the 16th Annual Meeting of*

the Special Interest Group on Discourse and Dialogue, pages 275–284, Prague, Czech Republic. Association for Computational Linguistics.

Wen, T.-H., Vandyke, D., Mrkšić, N., Gašić, M., Rojas-Barahona, L. M., Su, P.-H., Ultes, S., and Young, S. (2017). A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain. Association for Computational Linguistics.

Weng, L. (2021). Controllable neural text generation. *lilianweng.github.io/lil-log*.

Williams, J. and Young, S. (2007). Partially observable markov decision processes for spoken dialog systems. *Comput. Speech Lang.*, 21:393–422.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. (2019). Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Wu, C., Socher, R., and Xiong, C. (2019). Global-to-local memory pointer networks for task-oriented dialogue. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Wu, S. and Dredze, M. (2019). Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. In *EMNLP/IJCNLP*.

Xu, P. and Sarikaya, R. (2013). Convolutional neural network based triangular crf for joint intent detection and slot filling. In *2013 ieee workshop on automatic speech recognition and understanding*, pages 78–83. IEEE.

Xu, W., Haider, B., and Mansour, S. (2020). End-to-end slot alignment and recognition for cross-lingual NLU. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5052–5063, Online. Association for Computational Linguistics.

Yadav, V. and Bethard, S. (2018). A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. (2019). XLNet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*.

Yang, Z., Salakhutdinov, R., and Cohen, W. W. (2016). Transfer Learning for Sequence Tagging with Hierarchical Recurrent Networks. *CoRR*, abs/1703.06345.

Yang, Z., Salakhutdinov, R., and Cohen, W. W. (2017). Transfer Learning for Sequence Tagging with Hierarchical Recurrent Networks. In *ICLR*.

Yao, K., Peng, B., Zhang, Y., Yu, D., Zweig, G., and Shi, Y. (2014a). Spoken language understanding using long short-term memory neural networks. *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 189–194.

Yao, K., Peng, B., Zweig, G., Yu, D., Li, X., and Gao, F. (2014b). Recurrent conditional random field for language understanding. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4077–4081.

Yao, K., Zweig, G., Hwang, M.-Y., Shi, Y., and Yu, D. (2013). Recurrent neural networks for language understanding. In *INTERSPEECH*.

Yoo, K. M., Shin, Y., and Lee, S. (2019). Data augmentation for spoken language understanding via joint variational generation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7402–7409. AAAI Press.

Young, S., Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., and Yu, K. (2010). The hidden information state model: A practical framework for pomdp-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174.

Young, S. J. (2000). Probabilistic methods in spoken–dialogue systems. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 358(1769):1389–1402.

Yu, D., He, L., Zhang, Y., Du, X., Pasupat, P., and Li, Q. (2021). Few-shot intent classification and slot filling with retrieved examples. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies, pages 734–749, Online. Association for Computational Linguistics.

Zhang, X. and Wang, H. (2016). A joint model of intent determination and slot filling for spoken language understanding. In *IJCAI*.

Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., Gao, J., Liu, J., and Dolan, B. (2020). DIALOGPT : Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278, Online. Association for Computational Linguistics.

Zhang, Z., Zhang, Z., Chen, H., and Zhang, Z. (2019). A joint learning framework with bert for spoken language understanding. *IEEE Access*, 7:168849–168858.

Zhao, H., Yang, Y., Zhang, Q., and Si, L. (2018). Improve neural entity recognition via multi-task data selection and constrained decoding. In *NAACL-HLT*.

Zhao, Z., Zhu, S., and Yu, K. (2019). Data augmentation with atomic templates for spoken language understanding. In Inui, K., Jiang, J., Ng, V., and Wan, X., editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 3635–3641. Association for Computational Linguistics.

Zhong, V., Xiong, C., and Socher, R. (2018). Global-locally self-attentive dialogue state tracker. In *ACL*.

Zhong, Z., Zheng, L., Kang, G., Li, S., and Yang, Y. (2020). Random erasing data augmentation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 13001–13008. AAAI Press.

Zhou, H., Young, T., Huang, M., Zhao, H., Xu, J., and Zhu, X. (2018). Commonsense knowledge aware conversation generation with graph attention. In *IJCAI*.

Zhou, L., Gao, J., Li, D., and Shum, H. (2020). The design and implementation of xiaoice, an empathetic social chatbot. *Computational Linguistics*, Just Accepted:1–62.

Zhou, L. and Small, K. (2019). Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *ArXiv*, abs/1911.06192.

Appendix A

APPENDIX A.

	Method	Requirements	Model	Task
Transfer Learning				
Jaech et al. (2016)	MTL	Labeled conversational slot filling $\mathcal{D}_S$	biLSTM	SF
Goyal et al. (2018)	Pre-train & Fine-Tuning	Labeled conversational slot filling $\mathcal{D}_S$	biLSTM	SF,IC
Siddhant et al. (2019)	Pre-train & Fine-Tuning	Unlabeled conversational $\mathcal{D}_S$ Labeled conversational slot filling $\mathcal{D}_S$	biLSTM Elmo	SF,IC
Kim et al. (2017)	Expert model with attention	Labeled conversational slot filling $\mathcal{D}_S$	biLSTM + attention	SF, IC
Jha et al. (2018)	Expert model without attention	Coarse-grained labeled conversational slot filling $\mathcal{D}_S$	biLSTM	SF
Bapna et al. (2017)	Train on $\mathcal{D}_S$ only (zero-shot)	Labeled conversational slot filling $\mathcal{D}_S$ Natural language description of slot names	biLSTM	SF
Lee and Jha (2019)	Train on $\mathcal{D}_S$ only (zero-shot)	Labeled conversational slot filling $\mathcal{D}_S$ Natural language description of slot names	biLSTM+attention+highway	SF
Shah et al. (2019)	Train on $\mathcal{D}_S$ only (zero-shot)	Labeled conversational slot filling $\mathcal{D}_S$ Natural language description of slot names Slot value examples	biGRU	SF
Guerini et al. (2018)	Train on $\mathcal{D}_S$ only (zero-shot)	Slot value examples	biLSTM	SF
Liu et al. (2019c)	Train on $\mathcal{D}_S$ only (zero-shot)	Labeled conversational slot filling $\mathcal{D}_S$ Slot value examples	biLSTM+latent variable	SF
Data Augmentation				
Kurata et al. (2016b)	Model based DA	—	Se2Seq LSTM	SF
Hou et al. (2018b)	Rule, Model based DA	—	Seq2Seq + Attention	SF
Zhao et al. (2019)	Model based DA	Intent and slot Value Template	Seq2Seq LSTM	SF
Yoo et al. (2019)	Model based DA	—	Conditional Variational Auto Encoder	SF, IC
Peng et al. (2020)	Model based DA	Unlabeled conversational $\mathcal{D}_S$ Pre-trained GPT-2	Transformer	SF, IC
Anaby-Tavor et al. (2020)	Model based DA	Pre-trained GPT-2	Transformer	IC
Kumar et al. (2020)	Model based DA	Pre-trained BART, BERT, GPT-2	Transformer	IC

Table A.1: Comparison of transfer learning approaches for domain scaling

ATIS Slot	**OntoNotes Label**
AIRLINE_NAME	ORG
AIRPORT_NAME	FAC
ARRIVE_DATE, DAY_NAME, DAY_NUMBER, DEPART_DATE, DEPART_TIME, FLIGHT_DAYS, TIME_RELATIVE, TODAY_RELATIVE	DATE
ARRIVE_TIME, MONTH_NAME, PERIOD_OF_DAY, RETURN_TIME, TIME	TIME
CITY_NAME, FROM_LOC, STATE_CODE, STATE_NAME, STOP_LOC, TO_LOC	GPE
COST_RELATIVE, FARE_AMOUNT	MONEY
DAYS_CODE, ECONOMY, FARE_BASIS_CODE, FLIGHT_MOD, MEAL, MEAL_CODE, MEAL_DESCRIPTION, MOD, FLIGHT_STOP, FLIGHT_MOD, OR, RESTRICTION_CODE, ROUNDTRIP, TRANSPORT_TYPE	O
FLIGHT_NUMBER	CARDINAL

Table A.2: Label Mapping from ATIS to OntoNotes.

MIT Movie Slot	**OntoNotes Label**
CHARACTER, ACTOR, DIRECTOR	PER
YEAR	DATE
PLOT, RATING, TITLE, REVIEW, SONG, RATINGS_AVERAGE, GENRE, TRAILER	O

Table A.3: Label Mapping from MIT Movie to OntoNotes.

$\mathcal{D}_\mathcal{T}$	$\mathcal{D}_\mathcal{S}$						Avg	Δ
	TC	NW	BC	BN	WB	MZ		
TC	-	0.74	0.84	0.80	0.83	0.77	0.80	1.7
NW	0.74	-	0.85	0.91	0.91	0.90	0.86	0.7
BC	0.84	0.85	-	0.90	0.90	0.86	0.87	0.02

Table A.4: Domain Similarity (JSD) for each $\mathcal{D}_\mathcal{T}$ and $\mathcal{D}_\mathcal{S}$

Hyperparameter	**Value**
LSTM cell size	100
Dropout	0.5
Word embedding dimension	300
Character embedding dimension	100
Mini-batch size	32
Optimizer	Adam
Number of epoch	50
Early stopping	10

Table A.5: Neural Model Hyperparameters

Hyperparameter	**Value**
LSTM cell size	100
Dropout	0.5
Word embedding dimension	300
Character embedding dimension	100
Mini-batch size	128
Clip norm	1
Optimizer	Adam
Number of epoch	20
Early stopping	10

Table A.6: Neural model hyperparameters for MTL with Data Selection

Parameter	**Adopted value**
Surrogate model	Gaussian Processes with MCMC sampling
Acquisition function	Expected Logarithmic Improvement
Number of initial evaluation points	3
Search space upper bound	1
Search space lower bound	-1
Number of iterations	50

Table A.7: Parameters used by the Bayesian Optimizer.

Hyperparameter	Value
Learning rate	10^{-5}
Dropout	0.1
Mini-batch size	16
Optimizer	BertAdam
Number of epoch	30 (`bert-base-uncased`)
	10 (`bert-large`, `roberta-large`, `albert-xxl`)
Early stopping	10
nb_{aug}	Tuned on {2, 5, 10}
Nucleus sampling	top-p = 0.9
Max rotation	3
Max crop	3

Table A.8: Hyperparameters used for the Transformer based models and data augmentation methods

Dataset	#train
ATIS	7,846
SNIPS	24,472
FB	52,798

Table A.9: Total training examples for Slot-Sub-LM+Filter. The number of positive and negative examples are the same.

Dataset	Accuracy
ATIS	92.70
SNIPS	86.10
FB	99.14

Table A.10: The accuracy of the binary sentence classifier.

Model	**ATIS**		**SNIPS**		**FB**	
	Slot	Intent	Slot	Intent	Slot	Intent
BiLSTM + CRF	95.66	98.34	95.13	98.07	96.18	99.30

Table A.11: Slot filling and intent classification performance when 100% training data is used.

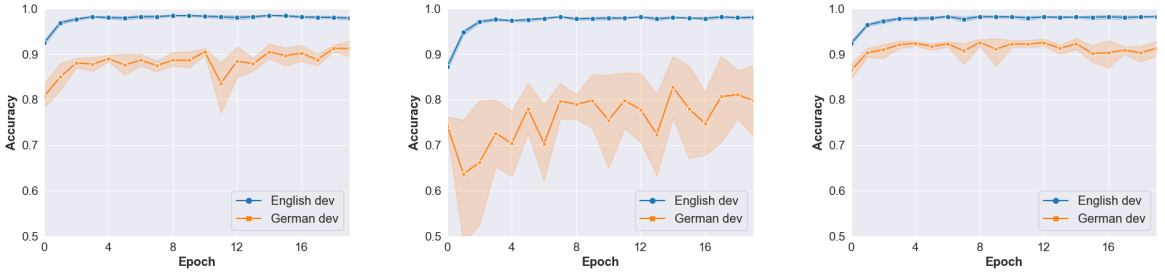
N	**ATIS**		**SNIPS**		**FB**	
	Slot	Intent	Slot	Intent	Slot	Intent
2	90.68	94.46	87.57	97.43	93.77	93.82
5	91.08	94.92	88.12	97.86	93.5	98.33
10	91.36	94.82	88.00	97.46	93.5	98.28
20	91.57	95.04	88.07	97.54	93.51	98.2
25	91.62	94.82	87.91	97.59	93.41	98.25

Table A.12: Slot filling and intent classification performance with SlotSub with different number of augmented sentence (N)

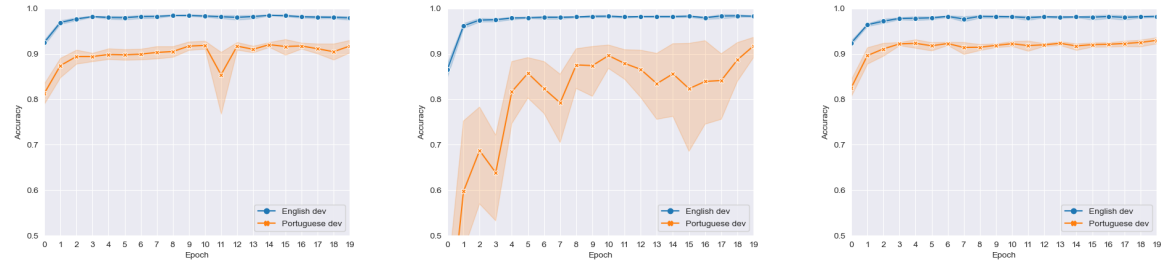
Dataset	Training Size (%)	p-value
ATIS-HI	5	0.04311444678
	10	0.005062032126
	20	0.04311444678
	40	0.04311444678
	80	0.1380107376
	100	0.2733216783
ATIS-TR	5	0.224915884
	10	0.005062032126
	20	0.7150006547
	40	0.1797124949
	80	0.1797124949
	100	0.1797124949
SNIPS-IT	5	0.04311444678
	10	0.005062032126
	20	0.04311444678
	40	0.04311444678
	80	0.04311444678
	100	0.04311444678
FB-ES	5	0.04311444678
	10	0.02831405495
	20	0.1797124949
	40	0.1755543028
	80	0.1380107376
	100	0.1797124949
FB-TH	5	0.04311444678
	10	0.005062032126
	20	0.1797124949
	40	0.1797124949
	80	0.1797124949

Dataset	Nb Aug	p-value
ATIS-TR	2	0.005062032126
	5	0.01251531869
	10	0.006910429808
	20	0.5001842571
	25	0.07961580146
ATIS-HI	2	0.1097446387
	5	0.005062032126
	10	0.005062032126
	20	0.04311444678
	25	0.04311444678
SNIPS-IT	2	0.005062032126
	5	0.005062032126
	10	0.005062032126
	20	0.04311444678
	25	0.04311444678
FB-ES	2	0.0663160313
	5	0.02831405495
	10	0.09260069782
	20	0.3452310718
	25	0.07961580146
FB-TH	2	0.03665792867
	5	0.005062032126
	10	0.005062032126
	20	0.04311444678
	25	0.04311444678

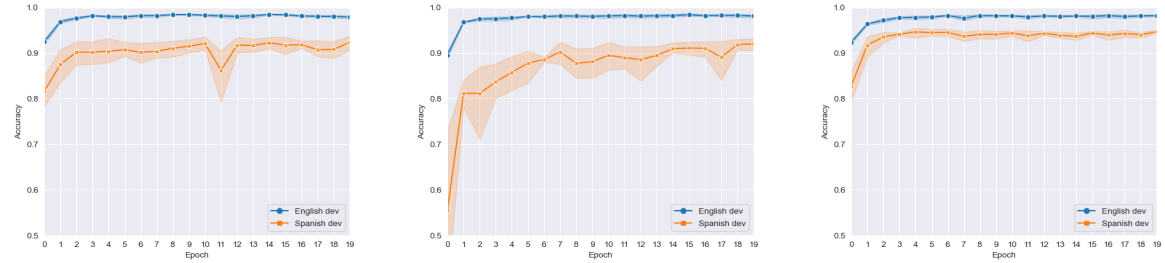
Table A.14: The p-values of statistical tests on the experiments



(a) No DAPT + FineTune-En (b) DAPT_{Tgt}+ FineTune-En (c) DAPT_{En}+ FineTune-En



(d) No DAPT + FineTune-En (e) DAPT_{Tgt}+ FineTune-En (f) DAPT_{En}+ FineTune-En



(g) No DAPT + FineTune-En (h) DAPT_{Tgt}+ FineTune-En (i) DAPT_{En}+ FineTune-En

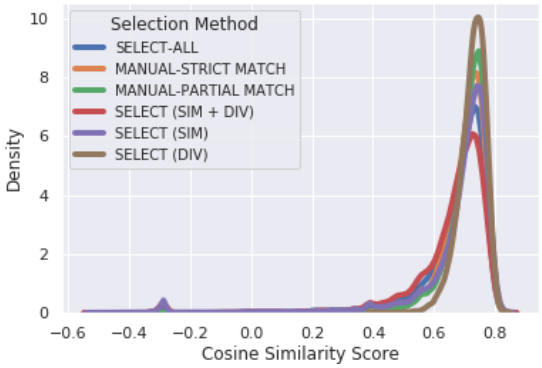
Figure A.1: Post-hoc analysis: *development set* performance variation across multiple runs on intent classification when using FineTune-En and applying different DAPT strategies.

A.1 Data Selection (DS) - Sentence Similarity

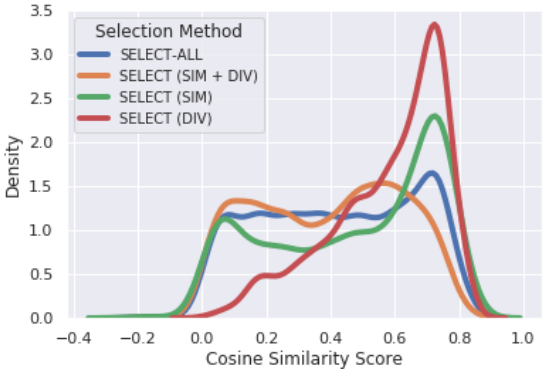
Next, we analyze the similarity distribution of the sentences *before* and *after* the data selection. We use Infersent encoder (Conneau et al., 2017) to obtain the sentence embeddings and using cosine measure to calculate the similarity score. The similarity score is calculated between each sentence embedding in $\mathcal{T}_S$ and the average values of sentence embedding in $\mathcal{T}_T$. We notice that the average similarity score distribution for methods without selection and with selection is similar, and that the standard deviation is very narrow. Observing the shape of the overall density[1] distribution of the similarity scores (Figure A.2), we can see that the distribution of DS_{all} is already highly concentrated to a particular area of similarity scores. Moreover, all the methods that use a data selection process produce similar shape of density distribution with respect to methods that do not use any selection process.

We hypothesize that in situations where the sentence similarity distribution is concentrated in a very narrow range, the benefit of performing data selection is lower. In order to validate if this is indeed the case, we created another source dataset, where the similarity score distribution is more uniformly distributed and includes data from different domains. We constructed this dataset by collecting the sentences from *all* sections of OntoNotes 5.0. After that, we computed the similarity of the sentences against the centroid of $\mathcal{T}_\mathcal{T}$. In order to generate a more uniform similarity distribution, we distributed the sentences into buckets of approximately the same size, where each bucket is defined by a similarity score range (e.g. 0.0 to 0.1, 0.1 to 0.2, etc.). We then carried on the same data selection evaluation using this new source data (we refer this as $\mathcal{T}_S^{all}$). We can see that after the selection process, the distribution tends to peak at a particular range (Figure A.2) again. Table A.16 shows the overall performance when using $\mathcal{T}_S^{all}$ data and we do not observe clear advantage of data selection (DS) methods. Based on the results, only on the MIT Restaurant dataset with DS_{sim} gives better performance than DS_{all}. Although DS_{all} still gives the best result for most datasets, among the three selection methods DS_{sim} consistently gives better results than the other two methods.

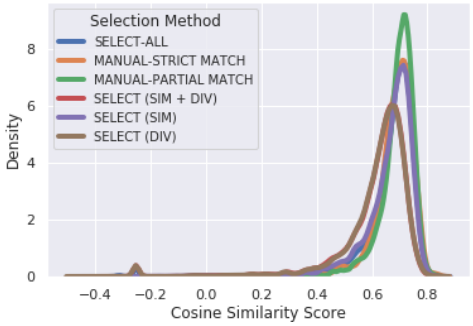
[1]Density plot is the smoothed version of the histogram estimated with Gaussian kernel density estimation.



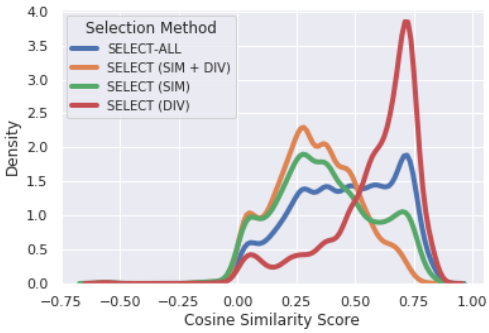
(a) ATIS with $\mathcal{T}_S$ as the auxiliary dataset



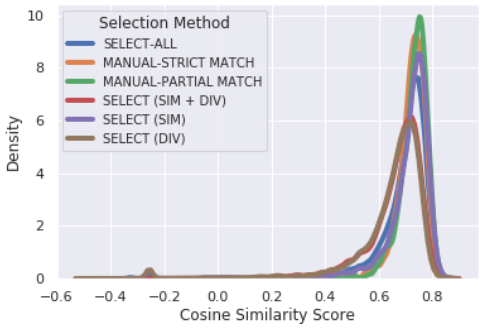
(b) ATIS with $\mathcal{T}_S^{all}$ as the auxiliary dataset



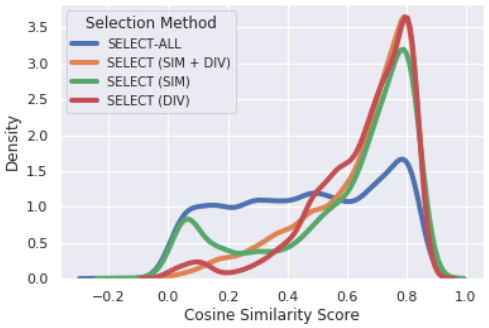
(c) MIT-R with $\mathcal{T}_S$ as the auxiliary dataset



(d) MIT-R with $\mathcal{T}_S^{all}$ as the auxiliary dataset



(e) MIT-M with $\mathcal{T}_S$ as the auxiliary dataset



(f) MIT-M with $\mathcal{T}_S^{all}$ as the auxiliary dataset

Figure A.2: Sentence similarity distribution when using $\mathcal{T}_S$ or $\mathcal{T}_S^{all}$x

OpenSub
You have a cancellation on Flight 16 for New York .
That route does not take us to the airport .
Chicago , this is flight 209er
I fly to Taiwan Tuesday then back to Dusseldorf
Your plane ticket to Naples .
I got 1,000 bucks that says you end up solo pilot on a one-way flight to Spankytown .
If we leave at 10 p.m. , we can catch them in Modesto .
Your bus leaves from depot six in just a few minutes .
do you know the way to san jose ?
He was boarding a flight to Johannesburg .
Take the 77 bus and get off at the village .
Anny flew here from Seattle today .
Just go back to their last port before she boarded , leave a message so Dean knows .
Flying monkeys , take us to Emerald City .
Will you sail from here or will you go via Rome ?
Where 's the plane , Francisco ?
Our estimated flight time is approximately five and a half hours .
Your flight 's in two hours .
So when you going to Cleveland ?

Table A.15: Example of the most similar sentences from OpenSub to the utterance in MultiATIS

.

Method	**ATIS**	**MIT-R**	**MIT-M**
DS_{all}	$\mathbf{91.14_{0.25}}$	$69.19_{0.45}$	$\mathbf{81.63_{0.25}}$
$\text{DS}_{sim,div}$	$90.66_{0.48}$	$68.88_{0.35}$	$81.24_{0.36}$
DS_{sim}	$90.88_{0.39}$	$\mathbf{69.45_{0.46}}$	$81.43_{0.34}$
DS_{div}	$90.77_{0.42}$	$69.01_{0.25}$	$81.25_{0.35}$

.

Table A.16: Results with $\mathcal{T}_S^{all}$ as the auxiliary dataset

A.2 Model, Implementation, and Training Details

- The infrastructure that we use is a machine with single GeForce RTX 2080 Ti GPU. The running time for performing continued pre-training with 100K sentences is around 45 minutes. The running time for the intent classification and slot filling experiments for five different seeds is around 100 minutes.

- For the intent and slot filling models, we adapt the implementation from Qin et al. (2020a) in which they make it publicly available (https://github.com/kodenii/CoSDA-ML). The sentence and token ratio replacement for code-switching is set to 1.0 and 0.9 respectively. For training, the learning rate is set to 10^{-5}, batch size is set to 32. We did not do extensive hyperparameter tuning, as this is a zero-shot cross lingual case where the target dataset is not available, we use the same hyperparameters as Xu et al. (2020).

- For the continued pre-training we use the language modeling script from Huggingface (Wolf et al., 2019). We use the `bert-base-multilingual-cased`, hidden state size is 768, we apply dropout probability of 0.1. The number of training steps is 12500 following Gururangan et al. (2020), the batch size is set