DIALOGUE STATE TRACKING FOR LOW-RESOURCE LANGUAGES

Hemanthage Supun Bhathiya

198128V

Thesis submitted in partial fulfillment of the requirements for the
Degree of Master of Science (Research) in Computer Science and Engineering

Department of Computer Science & Engineering

University of Moratuwa
Sri Lanka

July 2020

DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: Date:

The above candidate has carried out research for the Masters thesis/Dissertation under my supervision.

Name of the Supervisor : Dr.Uthayasanker Thayasivam
Signature of the Supervisor: Date:

ACKNOWLEDGEMENTS

I am sincerely grateful for the advice and guidance of my supervisor Dr.Uthayasanker Thayasivam. Without his help and encouragement this project would not have been completed. I would like to thank him for taking time out of his busy schedule to be available anytime that was needed with help and advice.

I would also like to thank my progress review committee, Dr.Surangika Ranathunga and Dr. Charith Chitraranjan. Their valuable insights and guidance helped me immensely.

I would like to thank the entire staff of the Department of Computer Science and Engineering, Both academic and non-academic for all their help during the course of this work and for providing me with the resources necessary to conduct my research.

Finally, I would like to express my gratitude to my family and all my friends for their support.

ABSTRACT

Dialogue State Tracking for Low-Resource Languages

Despite ground breaking work in the academia, current state-of-the-art work in goal-oriented conversational-agents has not been able to fulfil the demand of the industry for multi-domain multi-lingual, adaptable, dialogue systems. Data-intensive nature of deep learning models used in dialogue state tracking (DST) module, which is a core component of the goal-oriented dialogue architecture, and the lack of large labelled dialogue corpus for state tracking are two main factors which have hindered the progress.

We identified, modeling with separate natural language understanding (NLU) module and joint modeling of dialogue state tracker with NLU as the two main approaches for state tracking, and accordingly made two major contributions. First, we propose a novel meta-learning algorithm for intent detection and slot-filling tasks, focusing on models with separate NLU. Our work empirically demonstrates that the proposed meta-learning approach is capable of learning a meta-parameter(prior) from similar, but different tasks. Compared to the random initialization, which regular supervised learning algorithms rely on, proposed method significantly improves the accuracy in both intent detection and slot-filling tasks in few-shot (5-way 1-shot and 5-way 2-shot) settings. Further, our effective use of meta-learning for intent detection and slot-filling opens up new line of research for DST. Second, we systematically review the progression of joint NLU/DST models with special emphasis on their ability to generalize and adapt to new domains and languages.

Keywords: Dialogue State Tracking; Natural Language Understanding; Joint Intent Detection and Slot-Filling; Meta-Learning; Conversational-AI;

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
NLP	Natural Language Processing
NLU	Natural Language Understanding
DST	Dialogue State Tracking
DM	Dialogue Manager
NLP	Natural Language Generation
EQ	Emotional Quotient
IQ	Intelligence Quotient
RNN	Recurrent Neural Networks
CNN	Convolutional Neural Networks
LSTM	Long Short-Term Memory
NMT	Neural Machine Translation
NBT	Neural Belief Tracker

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

Decades of continuous efforts in academia to improve conversational Artificial Intelligence(AI) have resulted in several sub-disciplines. Based on the nature of the dialogue that the conversational-agent tries to engage, dialogue systems can be grouped into three major categories: question answering, social-chat and goal-oriented.[1]

Question Answering: QA agents are expected to provide direct answers to immediate user queries based on the knowledge acquired from various sources. Research in these bots are mostly focused on representation of the knowledge from different sources and little to no effort is made on modeling the context of the conversation.

Social Chat: These agents are expected to build a human-like conversation with the user. Similar to QA agents, these models may also depend on a knowledge base. However, modeling context of the conversation is an important aspect of these agents because the entire conversation with the user needs to be considered in generating a response. Further, these agents model the 'user' considering their intelligence quotient (IQ) as well as emotional quotient (EQ) in order to seamlessly connect with the user.

Goal-Oriented: Goal-oriented agents are expected to fulfil user tasks ranging from hotel or flight booking, bank transactions to scheduling meetings in a multi-turn conversation. The agent should be able to comprehend each user query within the context of the dialogue and engage in the conversation assisting the user to accomplish a particular goal.

Our work predominantly focuses on goal-oriented dialogues. With growing interest in the industry for replacing customer care operations with automated chat-bots, goal-oriented dialogues have been established as a separate sub-discipline, clearly differentiating from other types of conversational agents. Modular ar-

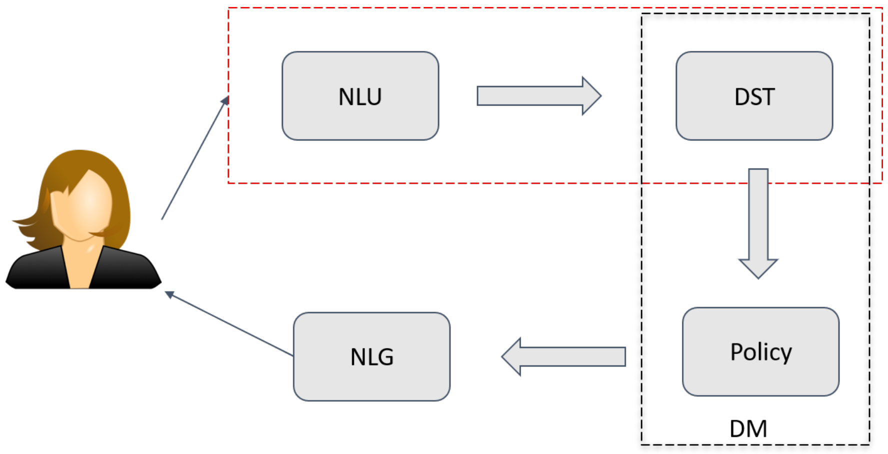


Figure 1.1: Goal-Oriented Dialogue System Architecture

chitecture and related natural language processing (NLP) techniques specific to goal-oriented agents have emerged.

1.1 Goal-oriented Dialogues : Architecture

This section will introduce the modules of goal-oriented dialogue systems which is essential in defining and understanding the scope of the work. We rely on a slightly evolved version of the [2] architecture which is shown in Figure 1.1. In high-level architectural perspectives, goal-oriented dialogue systems can be viewed as an integration of three main sub-modules:

Natural Language Understanding (NLU) : Takes user queries as input and converts into semantic form by performing intent detection and slot tagging.

Dialogue Management (DM) : Takes semantic representation of NLU as an input and acts as the controller of the system. DM can be further divided into two sub-modules:

- *Dialogue State Track (DST)* : determines the state of the system based on the semantic representation
- *Dialogue Policy Learning (DPL)* : determines an agent's actions based on the state. An action can be a response to a user or operation in a database.

Natural Language Generation (NLG) : Responsible for presenting actions selected by DM to user in natural language.

This work entirely focuses on natural language understanding and dialogue state tracking. Therefore we introduce related terminology and the expected functionality and the evaluation criteria in Section 2.1.

1.2 Motivation

Goal-oriented conversational AI, has been subjected to growing interest during the last decade with applications in a large number of industrial sectors. For instance, the hospitality industry deploys chat-bots (travel-bots) to facilitate tasks such as hotel booking and flight booking while the banking and finance sector utilizes chat-bots to support tasks such as checking account balance, money transfers and bill payments.

1.3 Problem Statement

Despite ground-breaking research in the field, a significant gap exists between current NLU capabilities of dialogue systems and the demand in the industry. Major causes can be listed as follows:

- With the effect of globalization, companies which are moving towards dialogue systems to provide their services, wish to support their clients, in their native language.
- The diversity of industry sectors that plan to adapt conversational agents is high. Further, some companies expect the same agent to fulfill needs within multiple domains. For instance, a company may needs to deploy an agent that supports booking a flight and a hotel while providing information to clients about attractions near the hotel. However, catering the demand for multilingual, multi domain support has been challenging due to the data-intensive nature of state-of-the-art deep learning models. Usually, these models need to be trained separately for each domain and each language. However, even for languages like English, availability of labelled dialogue corpus is limited. Companies being reluctant to release dialogue corpus due

to privacy concerns as well as tedious and complex labelling of dialogues have dampened efforts to build large corpus. In the case of low-resource languages, this barrier is even more challenging.

- Companies expect these dialogue systems to adapt to dynamic environments with frequent business policy and requirement changes. For instance, a hotelier which deploys a conversational agent may introduce new safety regulations due to COVID-19 which was unknown during the initial development. Current dialogue systems are limited in their ability to adapt to these unseen or unknown scenarios with. Developing dialogue systems with rapid inferring with only few examples is a challenge which requires special attention.

In summary, current state-of-the-art models are trained from scratch for each domain in each language and expect large amounts of data which is not not available in the real world.

1.4 Objective

The main objective of the proposed research is to:

- Evaluate and make recommendations on current models for DST
- Recommend mechanisms to utilize DST models for low-resource languages
- Develop DST models for low-resource settings with special focus on low-resource languages
- Evaluate DST models on low-resource settings

1.5 Contributions

- Novel meta-learning based approach for low-resource joint intent detection and slot-filling
- Comprehensive survey on the evolution of DST with respect to generalizability and adaptability aspects.

1.6 Organization

- Chapter 2 : This chapter consists of 3 sections which include background details necessary to understand this work. Section 2.1 introduces the problem of state tracking and the evaluation procedure with related metrics. Section 2.2 introduces several machine learning concepts which have been developed to tackle low-resource barrier associated with deep learning models. Section 2.3 explains the intuition behind meta-learning in detail.

- Chapter 3: Section 3.1 of the chapter focuses on research with separate NLU and discusses work related to problem of intent detection and slot-filling. Section 3.2 systematically study the evolution of joint models with special attention towards low-resource generalizability and adaptability aspects. Content of this section will be published as a review paper.

- Chapter 4: Introduces the proposed meta-learning model for few-shot joint intent detection and slot-filling, Meta-JOSFIN

- Chapter 5 : Provides details on experimental setup, results and discussion. We consider two separate experiments for evaluating our proposed algorithm:
  - Adaptability for unseen intents
  - Cross lingual adaptability

- Chapter 6: Contains the conclusion with probable future directions of the research

Chapter 2

BACKGROUND

2.1 Dialogue State Tracking

In-order to understand the functionality of dialogue state tracker , we will first introduce some terminology associated with goal-oriented dialogues.

Domain Ontology: Goal-oriented dialogues are associated with a predefined set of slots and list of probable slot-values for each slot which are collectively known as domain ontology. These slots are defined based on tasks that dialogue systems perform and information which needs to be gathered to achieve those tasks. Further, these slot-values are categorized as informable and requestable. User can provide a slot-value to the agent as a constraint for informable slots and request slot-values for requestable slots from the agent. For example, Dialog State Tracking Challenge 2 (DSTC 2) which competitors are asked to build dialogue agents for restaurant search define *pricerange* as a tracking slot with three possible slot-values: cheap, moderate and expensive. User can request the telephone number of a restaurant from the system while a search restaurant using the telephone number is not supported.

Dialogue State: Dialogue state s_t summarizes the dialogue up to turn t of a multi-turn dialogue, providing necessary information for the policy learner to choose the next (agent) action. This state includes, slot-values communicated by the user for different slots, requests made by the user on different slots, slot-values provided by the system to the user. Other than, predefined set of probable slot-values, slot-value may be denoted as *None* if the slot-value is not given by the user up to that turn or *dontcare* if user has mention in the dialogue that (s)he don't want to make any constraint based on a particular slot. In the DSTC2 restaurant example, conversation would start with None for *pricerange* and if the user said (s)he never minds the price range of the restaurant, it will be denoted as *dontcare*.

More formally, state can be defined in terms of :

- *Goals*: True value of all the slots defined in the ontology as mentioned up to the current turn
- *Method*:Way that the user expects to interact with the system. In DSTC2&3 values can be by name, by constraints, by alternatives or finished and may be inferred from the user-acts. Some other dialogue systems may prefer all user-acts itself as part of the dialogue state.
- *Requested slots* List of slots requested by user up to current turn which are not yet provided by the system.

Dialogue State Tracker: State tracker is expected take all the observable inputs up to the time t including user queries, system actions with resulted outputs and return the true current state $s*$ in that time. However, the true state is not observable due to reasons such as issues in speech recognition (in spoken dialogue systems), errors in language understanding, ambiguous user queries, non-communicated changes in user goals. Therefore, dialogue state trackers usually output belief estimates for multiple states (distribution over possible dialogue state). Belief estimate is preferred over the single most probable state as it provides a better insight for dialogue-policy-learner to make clarification actions.
More formally tracker output can be defined as:

- *Goals*: For each informable slot, tracker output the distribution over probable slot-values. Joint goal distribution may be reported if the goal is not made of independent slot distribution.
- *Method*:Distribution over methods defined in the ontology.
- *Requested slots* For all requestable slots in the dialogue, this defines the binary distribution whether the user has requested and the system has to inform.

Figures 2.1 illustrates step by step function of dialogue state tracker. However, it should be noted that both domain ontology and the belief states are internal parts of DST. Further, the diagram only shows a single value for a particular slot. But state trackers usually output a distribution over all the possible slot-values.

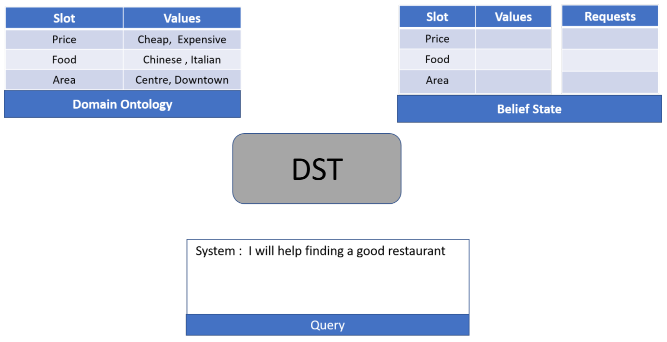
2.1.1 Datasets

In this section, we introduce a few very commonly used benchmark datasets in detail. However, we do not cover all the available datasets. For interested readers, [3] provides a comprehensive survey of available datasets (as of 2018) covering all the main conversational AI tasks including DST. Other than listed datasets in above work, the annual DSTC (initiated as Dialogue State Tracking Challenge and renamed to Dialogue System Technology Challenge) competition series (currently in 9th edition) provide useful data sets for state tracking research.

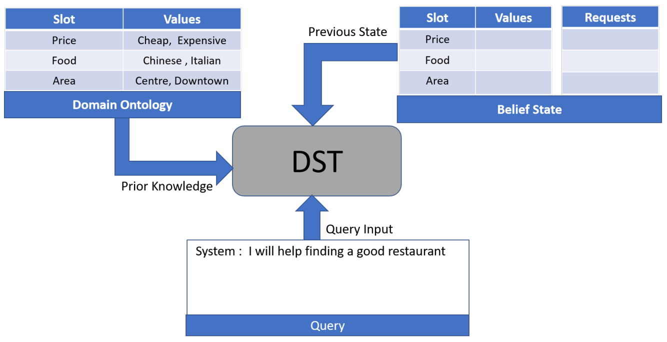
DSTC1 [4] : This corpus contains dialogue data of passengers who enquires route information from a city bus company over the phone. For example, a passenger may ask the time that the next bus leaves from a particular location to his intended destination. Ontology is defined with 5 main slot types (route, from, to, date and time). Both from and to slots were subdivided to three categories: desc, neighbourhood, monument (such as from.desc or to desc) which make a total of 9 slots. Training dataset is divided into 4 training sets based on how data is collected. Four test sets are based on the similarity to various training datasets. Presenting multiple test sets with different levels of similarities with training datasets facilitate testing generalizability of dialogues systems built.

DSTC2 [5]: DSTC2 corpus contains dialogues in the restaurant information domain. Corpus contains a train set with 1612 calls, a dev set with 506 calls and a test set with 1117 dialogues. Note that all the slots are requestable while some of them are informable as well with predefined values.

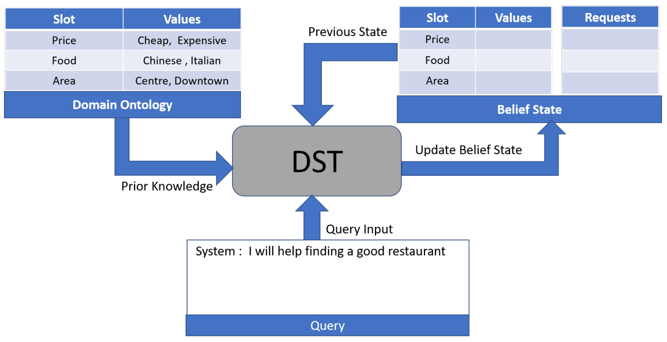
DSTC3 [5] : DSTC3 is an extension of DSTC2 towards a more complex tourist information domain and ontology defines more values under available slots while introducing more slots. However, DSTC3 labelled dataset is smaller (with only 10 dialogues) compared to DSTC2. Purpose of DSTC3 is to test generalizability aspects of models (in terms of domain adaptability and expandability, learnability



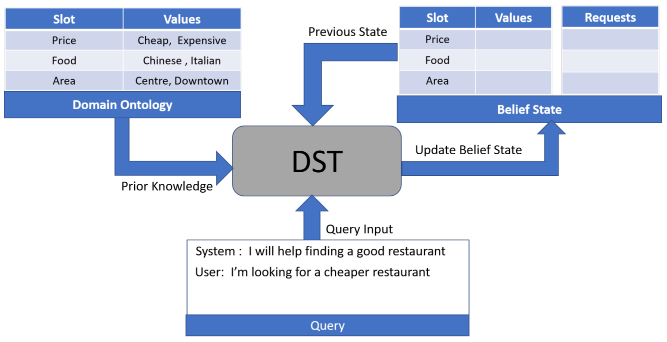
(a) DST components



(b) takes user queries, domain ontology and previous state for processing

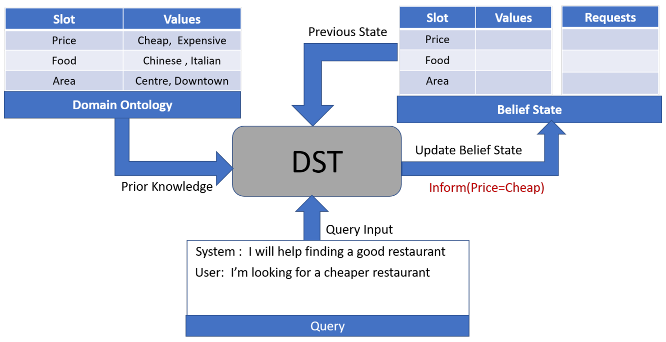


(c) send output to the belief state after processing



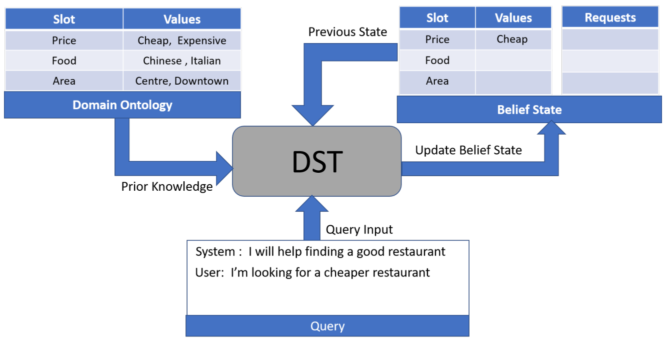
(d) User query "I am looking for a cheaper restaurant"

Figure 2.1: Dialogue State Tracker :Functionality



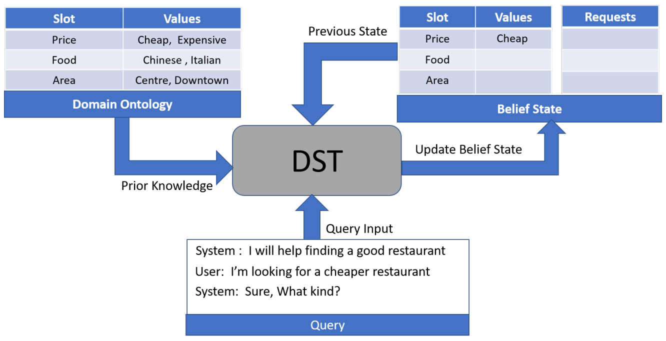
System : I will help finding a good restaurant
User: I'm looking for a cheaper restaurant

(e) Process the query and pass the dialogue act "Inform(Price=Cheap)"



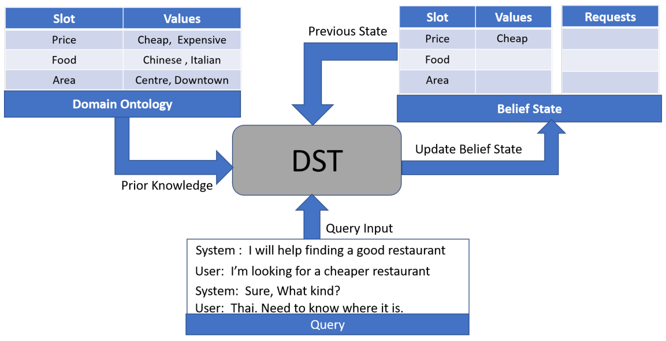
System : I will help finding a good restaurant
User: I'm looking for a cheaper restaurant

(f) Slot *price* in belief state is updated with value *cheap*



System : I will help finding a good restaurant
User: I'm looking for a cheaper restaurant
System: Sure, What kind?

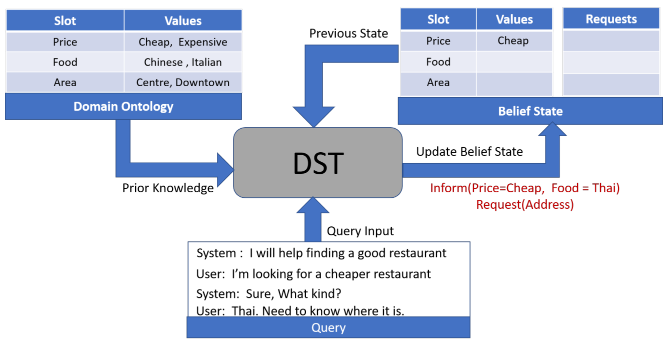
(g) System respond to user "Sure, What Kind?" (This involves DPL and NLG)



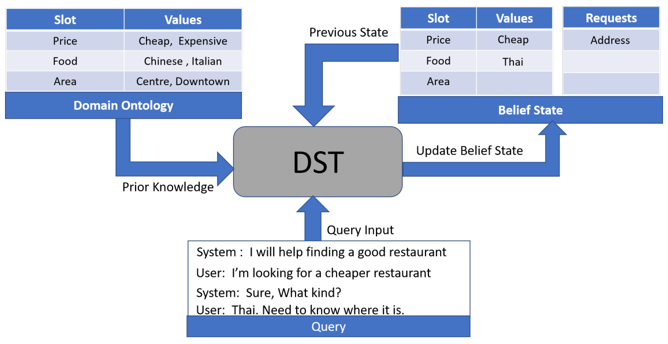
System : I will help finding a good restaurant
User: I'm looking for a cheaper restaurant
System: Sure, What kind?
User: Thai. Need to know where it is.

(h) User respond, "Thai. Need to know where it is"

Figure 2.1: Dialogue State Tracker :Functionality



(i) pass the dialogue act "Inform(Price=Cheap,Food=Thai), Request(Address)"



(j) Corresponding slots updated in belief state

Figure 2.1: Dialogue State Tracker :Functionality

of unseen and unknown slots) made for DSTC2.

WoZ2.0 [6] : Similar to DSTC2, WoZ2.0 ontology is defined within the restaurant domain where dialogue assists users to search for a restaurant in Cambridge UK. Dataset contains about 600 dialogues (with some dialogues being unfinished) within restaurant domain. There are 6 slots in ontology altogether: food, pricerange, area, address, phone, postcode with only the first 3 slots being informable.

Multi-WoZ [7] : This corpus contains a total of 10438 dialogues (with 3406 single domain and 7032 multi-domain dialogues) dialogues across 7 domains; attraction, hospital, police, hotel, restaurant, taxi, train. These domains are naturally connected within a dialogue with single dialogue span up to 5 domains. For example, a tourist would find a hotel, book a train and a taxi to reach the hotel and search for attractions near the hotel. This dataset has been extensively used in many multi-domain DST research. Multi-WoZ is magnitude larger and complex compared to previously mentioned datasets.

2.1.2 Evaluation Metrics

Single metrics is unlikely to evaluate the performance of a dialogue system due to its inherent complexity. Therefore, DSTC2&3 propose several different metrics each measuring different aspects of the dialogues. Out of these metrics, 3 metrics are featured among competitors [16]:

- *Accuracy*: measure of 1-best quality
- *L2 norm:* probability calibration measure
- *ROC Correct Accept (CA) 5*: This is the fraction of correct accepts when there are at most 5% false accepts (FAs). Assess the discrimination.

It should be noted that each of the above metrics are used to calculate goals, methods and requested slots separately. However, most the recent DST research tends to compare model performance based on:

- *Joint Goal Accuracy*: Compares a predicted state at turn t against the ground truth at that turn. Output is considered correct, if and only if all values of the prediction exactly match the ground truth.
- *Slot Accuracy*: Individually compares *(slot, value)* pairs, or *(domain, slot, value)* triplets in multi-domain dialogues, with the ground truth.

2.2 Low-Resource Deep Learning and NLP

Demanding large amounts of data is characteristic to almost all the deep learning models regardless of the domain applied. In this section, we will look into machine learning techniques which have emerged to overcome this resource barrier. Two most successful techniques which have been used to adapt deep learning models for low-resource settings are:

- Transfer Learning

- Meta Learning

Early works in both of these techniques have proven successful in computer vision and image processing domains and then applied in NLP tasks.

2.2.1 Transfer Learning

[8] defined transfer learning as: given a source domain D_S and learning task T_S, a target domain D_T and learning task T_T , transfer learning aims to help improve the learning of the target predictive function f_T in D_T using the knowledge in D_S and T_S, where $D_S = D_T$, or $T_S = T_T$

Further [8] lists 4 different types of transfer learning approaches:

Instance-transfer: Re-using knowledge from the source domain to the target task. However, all the source domain examples may not be equally important. Therefore source instance relatedness to the target is re-weighted.

Feature-representation-transfer: Discover a set of good feature-representation which reflects the common characteristics of both source and target domain. This minimizes domain divergence and reduce error rates in classification and regression models.

Parameter-transfer: Finds shared parameters or priors between source domain and the target domain models. These shared params enable cross domains transferring.

Relational-knowledge-transfer: Maps relational knowledge between the source domain and the target domains. Both domains are relational domains and Independent and identically distributed (i.i.d) assumption is relaxed in each domain

Based on different settings, transfer learning can be categorized into three:

Inductive Transfer Learning: Source and the target domains are the same but the tasks within are different. Transfer learning attempts to get insight from the inductive biases of the source domain to perform well on the target domain.

Unsupervised Transfer Learning : Similar to inductive transfer, but the focus is on unsupervised target tasks

Transductive Transfer Learning : Source and the target domains are different but there are similarities between corresponding tasks. This setting expects a large amount of label data for the source domain with no labelled data in target domain.

Transfer Learning in Deep Learning: Previous introduction and categorizations of transfer learning are generally applicable to all the machine learning techniques. This subsection will look specifically into transfer learning in the context of deep learning.

Training deep learning models is an example for an inductive learning process. In inductive learning, models infer mapping from a set of training examples. For instance, an image classification model learns a mapping between image features and image class labels. Set of assumptions that models make on distribution of the training data in order to generalize for unseen scenarios is known as inductive bias. These biases affect what the model learns.

Inductive transfer learning techniques make use of inductive biases of source tasks to facilitate target tasks. These facilitating could be of different forms such as narrowing the hypothesis space or adjusting the search process itself.
Several deep transfer learning strategies have emerged:

Off-the-shelf pre-trained models have emerged as a successful way of deep transfer learning. Deep learning models being layered architectures which different features are learnt in different layers, idea is to leverage weighted layers from a pre-trained model. Depending on the task these weights may be:

- Freezed : weights not updated in training process
- Fine tuned : rather than freezing all the layers, some layers, which learn generic features, are freezed while other layers, which are more task specific, are fine tuned.

There are few well known pre-trained models in:
Computer Vision : VGG-16, VGG-19, Inception V3, XCeption, ResNet-50
Natural Language Processing : Word2Vec, Glove, FastText , MUSE, Bert

Due to their relatedness and importance to our work, we also introduce some of the NLP related pre-trained models in this section.

Word2Vec : Earliest model to represent words in a continuous vector space. [] proposed two algorithms namely; *skip-gram* model and *continuous bag of words* (CBOW) model. Both models are architecturally similar, but the CBOW model predicts the current word based on the context while the skip-gram model predicts the context given the current word. Representations learnt from skip-gram model have outperformed the CBOW based representations on many similarity tasks

Glove: Global vector representation combines context window approaches like skip-gram with global matrix factorization. Glove model utilizes statistical information by training only on the nonzero elements in a word-word co-occurrence matrix, rather than on the entire sparse matrix or on individual context windows in a large corpus.

FastText: Consider character n-grams opposed to the Word2Vec which learns vectors only for complete words found in the training corpus. For example the word undo will be represented as the sum of all n-grams <un,nd,do,und,ndo> given the hyperparameter minimum n-gram size is 2. Use of n-grams enables FastText to generate better embeddings for rare words as well as out-of vocabulary words.

MUSE: All the prior word vectorization mechanisms discussed, usually represent words within a single language in a continuous vector space. Multilingual Unsupervised and Supervised Embeddings (MUSE) align representations in different spaces to a single continuous vector space such that the cross-lingual relationships are encoded while intra-language relationships between words are preserved.

BERT: Bidirectional Encoder Representations from Transformers (BERT) model is known as contextual word embedding, meaning context data is incorporated with word representation.For instance, character sequence “mouse” would mean entirely different things based on the context and BERT would have two separate representations. Compared to aforementioned models where embedding layer is usually freezed when used for different tasks, BERT representation would be fine-tuned.

2.3 Meta Learning

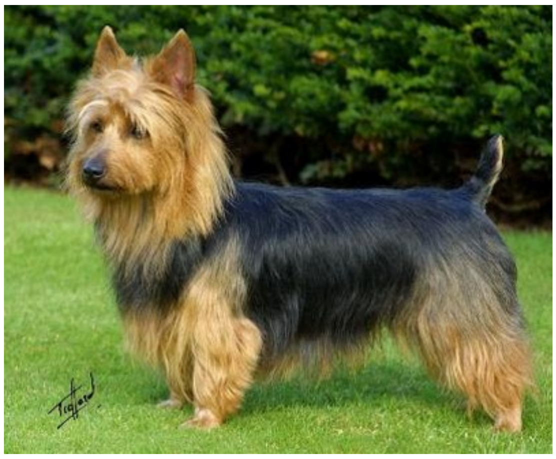
Meta-learning or *learning to learn* refers to the process of systematically observing the way machine learning models perform on different tasks and using that experience to learn new tasks or skills. This paradigm is motivated by the fact that human brain never learn skills from scratch, rather infer from past experience in similar(but different) settings. Objective of meta-learning is to enable machine learning models to rapidly infer on new tasks utilizing prior knowledge gained from similar experiences.

Defining the similarity between two tasks is a separate research question which we will not discuss in this work. However, intuitively we consider two tasks are different but similar when there is a common underlying skill for both the tasks or the same task is performed in a different environment. For example running and walking on a jogging track are two different tasks demanding lots of common characteristics. On the other hand, walking on a road differs from walking on a jogging track therefore they can also be considered as two different (but similar) tasks.

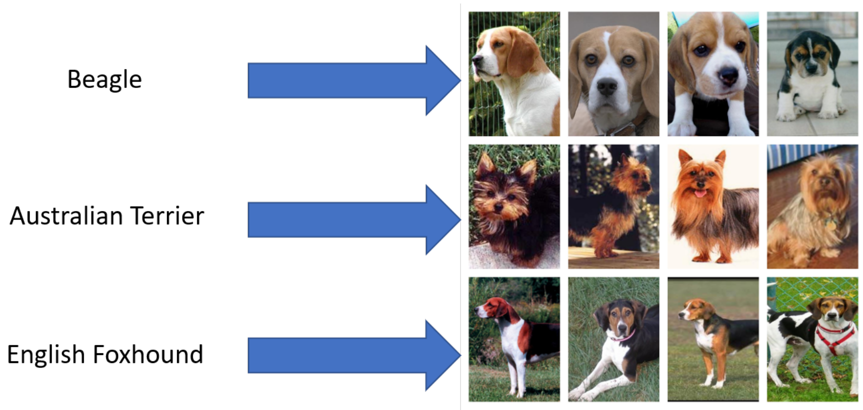
2.3.1 Meta-Learning for Deep Neural Networks : Intuition

There are two schools of thoughts defining what is considered as a meta-learning problem. Some scholars view any process which learns from data related to training (meta-data) as a meta-learning process. This definition of meta-learning covers research areas such as hyper-parameter tuning, Auto-ML. However, our work will use the term 'meta-learning' in a narrower sense, mostly focusing on neural approaches, especially deep learning. Recently meta-learning and deep learning combination has been a hot topic in the area of few-shot image recognition. We will describe the intuition behind meta-learning using an image classification problem.

Question 1:*Imagine your know absolutely nothing about dog breeds and answer* *the question What is the breed of this dog?* A) Beagle B) Australian Terrier C) English Foxhound



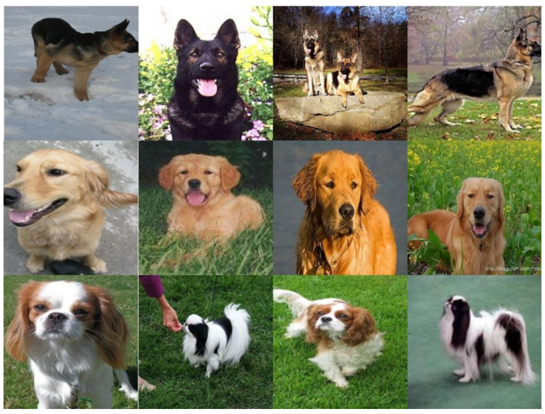
Question 2: *What if following examples are provided and asked the question 1?*



Almost all the human will be able to answer question 2 even if he or she had no knowledge of the breed of the dog prior to that. This is because the human brain has the ability to infer rapidly and differentiate breeds with only a few examples. Our problem of interest can be summarized as *"How can we train a machine learning model to do what we (our brain) did in question 2?"*

It may occur that the above example is just a classification problem and standard supervised learning would easily solve the problem. However, this leads to two issues. Firstly, we only have very few examples for each breed and image classification models expect large amounts of data to form an accurate model. Second, and perhaps the most important issue can be understood from Question 3.

Question 3: *Will the supervised learning model trained for prior example be able to accurately classify the following example?*



Answer to Question 3 is, the supervised learning models need to be re-trained from scratch to classify new set of breeds. This illustrates the difference between skills that supervised learning models acquired compared to human skills. We, human have the understanding of the concept of 'breed' as opposed to the label names remembered by normal supervised learning models. Meta-learning aims to gain this skill.

Few-Shot Learning: As a side note we will also use the prior example to introduce the idea of few-shot learning and related terminology. Few-shot learning refers to the practice of training machine learning models with very small amounts of data. Term, k-shot is a spin-off of few-shot learning where the k is used to denote the number examples used. zero-shot, one-shot with explicit k are commonly used. k-shot learning has been widely used in multi-class classification problems. Multi class classification among n different classes is known as n-way classification. Therefore, multi-class classification problems within a few-shot learning setup are commonly referred to as n-way k-shot classification. In n-way k-shot setting, model is expected to learn n-way classification with only k*n examples with k examples under each intent. question 2 and 3 are examples of 3-way (3 types of breeds) 4-shot (4 examples for each breed) classification problem.

2.3.2 Transfer Learning vs Meta Learning

Transfer learning and meta-learning differ in the way the models are trained. In transfer learning, the model is optimized in the data-space of a particular source task with a large dataset and learnt model parameters are used for low-resource target tasks. Meta-learning algorithms in contrast, are meta-trained in task-space with the focus of learning an update rule or generalized parameterization without optimizing for a particular task. It is during the adaptation phase that meta-learning algorithms are tuned for specific tasks.

2.3.3 Related Work in Meta Learning

Prior work in learning to learn paradigms date back to [9] where genetic programming approach is taken. [10] was mostly inspired by the behavioural difference between biological neural networks and artificial neural networks. However, it is [11] paper which critically questioned the generalizability of machine learning models in contrast to human learning ability, motivated the deep learning community to revisit these fundamental concepts learning to learn. The paper also proposed a probabilistic model with Bayesian programming learning which outperformed many deep learning models in one-shot classification. Following this success in few-shot classifications tasks, [12] proposed a meta-learning model using neural networks with external memory capacities for few-shot classification. [13] extended the trend for generative models with few-shot image generation. Other significant works include MAML [14] with optimization based inference approach to find a prior, SNAIL [15] with black box adaptation approach and [16] which took a non-parametric approach. Going beyond supervised learning setups, [17] has shown, RL tasks can also benefit from meta-learning.

Even though early research in the intersection of meta-learning and deep learning was limlited to computer vision, recent work has shown that many NLP tasks can benefit from this paradigm. [18] proposed a meta-learning approach for low-resource neural machine translation (MetaNMT) with the support of unified lexical representation. [19] investigated different meta-learning algorithms for low-resource NLU. [20] proposed a meta-learning approach for low-resource NLG in goal-oriented dialogues. [21] used meta-learning for structured query generation. Common characteristic of most the aforementioned research is the adaptation of the MAML algorithm. This popularity of MAML is mainly due to its simple and general nature with no constraint on the model architecture or loss function. Further, most of these work cast problems to set-to-set few-shot learning setting.

Chapter 3

RELATED WORK

In broader sense, dialogue state tracking models can be classified under two categories.

- Models with separate NLU module
- Models with joint NLU/DST

In first type of models, NLU takes single user utterance and performs tasks such as domain detection, intent detection and slot-filling. Results from these individual utterances based NLU tasks are passed to dialogue state tracker where they are combined with the context details for resolving the state. In contrast, joints models take user utterance directly as an input and perform state tracking. Generally, joint NLU/DST models have outperformed separate NLU models in recent research.

3.1 Models With Separate NLU

Despite the success in joint models in comparatively high-resource settings, we believe that further research is necessary to decide which level of integration performs well under low-resource settings. It is commonly accepted that the amount of data required increases when more and more components are combined. Therefore, we will look into the NLU module which is a core-component in goal-oriented dialogue systems which semantically represent individual user queries.

3.1.1 Intent Detection and Slot-Filing

The major role of NLU components is to identify the intention (intent detection) and extract attribute-values conveyed (slot-filling) from user utterances. 3.1.1 shows a sample user utterance with detected intent and slot tags following the

IOB representation. (slot,value) pairs such as *(datetime, latest)* and *(location, New York)* are extracted from the utterance.

Utterance	Slot Tags	Intent
give	O	find-weather
your	O	
latest	B-datetime	
weather	B-weather	
report	I-weather	
for	O	
New	B-location	
York	I-location	

Table 3.1: slot tags and intent of the sample utterance *"give your latest weather report for New York"*

Problem of slot-filling is usually modeled as a sequence labelling problem. Given an input sequence $X = \{x^1, x^2 ... x^n\}$, system outputs $Y = \{y^1, y^2 ... y^m\}$ where y^i is chosen from predefined set of tags $\mathscr{Y}$. So the problem can be defined formally as follows: given $((x(n), y(n)) : n = 1, ..., n)$ find a function $f : X -> Y$ where $y^i \epsilon \mathscr{Y}$. Intent detection on the other hand is considered a classification problem. Given an input sequence (a user query) and set of possible intents $\mathscr{I}$, system classifies the intention of the user. Traditionally, separate models were developed for each of these tasks. However, recent models, try to achieve these two tasks in a joint model due to their intimate relatedness. [22] propose few key changes to the common encoder decoder architecture. First, bidirectional LSTM is used as the recurrent unit of the encoder. Second, alignment details of sequence input and the sequence output are incorporated explicitly, rather than model expecting model to learn this relationship. Third, attention mechanism which resolve the issue of forgetting earlier parts of utterance. Attention, allows model to automatically (soft-)search for parts of a source sentence which are relevant for tasks. We adapt this attention based encoder-decoder model as our *base-model* for joint intent detection and slot-filling.

3.1.2 Related Work in Intent Detection and Slot-Filling

Early methods in intent detection includes SVMs [23] and Boosting [24]. These models relied on manually extracted features and failed to capture the deep semantics of user queries. With the success of deep learning, scholars started to utilize word embedding for intent detection tasks [25, 26]. Being able to represent semantic relatedness between words, these models outperformed traditional approaches.

Motivated by the success in Convolution Neural Network (CNN) in the image processing domain, [27] experimented CNN for text classification, achieving state-of-the-art results. However, CNN capabilities were limited due to its inability to grasp sequential nature of text. [28] introduced RNNs for intent detection with significant reduction in error rate. [29] replaced vanilla RNN units with LSTMs to overcome the gradient explosion and vanishing problems.

Before deep learning techniques gain traction, CRFs [30] and SVMs [31] were used for slot-filling. [32] introduced RNN based approach for problem of slot-filling with state-of-the-art results on ATIS (Airline Travel Information Systems) benchmark.

Even though initial research considered slot-filling and intent detection as two separate tasks and built models for each task accordingly, recent works tends to achieve both tasks in a single model. [33] proposed a unified framework where pipeline approach was taken. Results from slot-filling were used as a feature for intent detection. [34] have shown that models which optimize two tasks simultaneously outperform mono-objective models as well as pipe-lined cascade models. [22] introduced attention mechanism to learn a soft alignment and capture important semantic components.

3.1.3 Related Work in Low-Resource NLP

Early work in NLP, [35] relied on rule based approaches. Language specific rules were defined and separate models had to be designed for each and every language regardless of the amount of data available. Deep learning models in

NLP mentioned earlier on the other hand relied on a vector representation learnt from the data itself. Same model architecture could be used for many languages. However, these models require large data corpus for training. Transfer learning and meta-learning are two main lines of research adapted to overcome this resource barrier.

Following the success of transfer learning [36] in image processing , [37] applied transfer learning for document classification and sentiment analysis. [38] successfully applied transfer learning for Neural Machined Translation(NMT). More recently, transfer learning has been used for slot-filling [39]. Transfer learning utilize the model parameters learned from source-task with large data corpus to improve performance on similar target-tasks with little or no training data. Our proposed approach focuses on more constrained setting where large corpus of similar single task is not available. Datasets we used, to meta-learn initializing parameter is smaller in magnitude compared to datasets where transfer learning approaches were proven successful.

3.2 Evolution of Joint NLU/DST Models

This section reviews the evolution of join DST/NLU models with an emphasis on the generalizability and adaptability within data scarce settings. Content of this section will be published in a journal.

3.2.1 Use of RNN and De-lexicalization

[40] is an early attempt in joint modeling of DST. The proposed model, which focuses on spoken dialogue systems, feeds automatic speech recognition(ASR) hypothesis directly into state tracker omitting spoken language understanding (SLU) processing steps. Proposed word-based approach avoids the necessity for engineering feature function to build a semantic representation. Instead of few chosen features, the suggested model uses n-grams extracted from ASR N-best list combined with the last machine act. Model achieves the generalizability for unseen states by introducing 'tagged' features. In parallel to learning specific examples

with extracted n-grams (such as *like indian food*, *serve indian food*) model also learns to generalized representation by tagging n-grams (*like <tagged-value> food*, *serve <tagged-value> food*). Intuition behind the mechanism is, in a sentence with n-gram <unseen_value> food, it is likely that $foodtype =< unseen\_value >$. For each slot ($s_l$), these high-dimensional features are extracted and fed into an RNN with probable slot-value (s_v) for that particular slot. Model outputs the probability distribution over the set of probable slot-values along with *None*.

[41] proposed a multi-domain DST model built up on previously mentioned RNN architecture. Compared to the original model in which only tagged possible slot-values in the n-gram and used as a feature, this model delexicalize both slot-values as well as slot-names. Delexicalized version of *like Indian food* will be *like <tagged-value> <tagged–name>*. Features with this level of delexicalizing enable transfer learning between unseen values as well as entirely new slots. This ability to learn new slots consequently help learnability in new domains. Further, in-order to achieve domain-independence, the model get rid of a component in the original RNN which learnt a mapping from untagged n-grams to specific slot-values. Transfer learning process work as Algorithm 1:

Algorithm 1 Transfer Learning Process in [41]

```
 1: while Learn Generic Initialize Parameter (General Dialog Learning): do
 2:     for each domain do
 3:         Replace all slot-name, slot-value occurrences with a generic tag (delexi-
            calize)
 4:         Tie all the RNN parameters across each slot.
 5:         Train the shared RNN model parameters with slot-agnostic delexicalized
            dialogues
 6:     end for
 7: end while
 8: while Slot Specific Learning: do
 9:     Initialized slot-specialized model with shared RNN model parameters
10:     Perform further training with slot-specific delexicalized training data.
11: end while
```

3.2.2 Complete Neural Approach

Despite aforementioned early attempts motivated by the success in deep learning, it is Neural Belief Tracker (NBT) [42] that leveraged the potential of deep learning for DST to an unprecedented level. [41] has shown that semantic dictionaries are essential for exact-matching, delexicalization based models to perform well which hinder their adaptation in real world domains. NBT model overcomes this challenge by effective utilization of semantic information encoded within pre-trained word vectors. For example, exact-matching dialogue system would depend on a predefined dictionary to know *inexpensive* is equivalent to *cheap* where the proposed model would capture this similarity from the positioning of respective word vectors.

Roles of each component can be listed as below:

User Utterance Representation: Paper proposed two learning representation models NBT-DNN and CNN based NBT-CNN which significantly influenced the direction of DST research. NBT-DNN model takes cumulative sums of all n-grams where n=1,2,3 scenarios. Then each category of n-grams is passed through a non-linear layer separately before the results of all categories are summed to obtain final utterance representation. NBT-CNN model introduces CNN layers with window sizes 1,2,3 which filters uni-grams, bi-grams and tri-grams. CNN layer is followed by ReLu and maxpool layers before summing to obtain final representation of the utterance.

Semantic Decoding: Interacts (slot, value) pair candidate representation (c) with user utterance representation (r) to decide whether user has explicitly mentioned the value in the last user turn of the dialogue. This decoding does not take the context of the dialogue into account. It should be noted that the decoder outputs element-wise vector multiplication as size of DSTC2 and WOZ did not suffice to result in meaningful representation from the interaction between r and c vectors.

Context Modelling: Understanding of the context is essential to extract information in multi-turn dialogues. System acts, requests and confirms are of vital importance in decoding some user queries. Requests and confirms are two common system acts in dialogues which are vital in the extraction process. NBT model captures this system act information under the Markov assumption that the last system acts matter. User utterance representation(r) is directly fed as an input while context representation (t) is passed through the gating mechanism only if it is mentioned about current slot,slot-value pair.

Binary Decision Making: Results from semantic decoding and context modeling are passed as inputs to output binary softmax, which indicates whether a given slot-value is provided or not.

Even though proposed NBT takes a fully data-driven approach when deciding informable and requestable slots at given user utterance turn t (based on system act at $t-1$), a major drawback of this work is the rule-based approach used when deciding whether to update the belief state or not. This rule-based mechanism relies on a hyper-parameter which requires tuning for each new domain or environment the model is deployed. [43] proposes two statistical update mechanisms to overcome these drawbacks in the original NBT model. In the first mechanism named *One-Step Markovian Update*, current belief state b_t^s for a particular slot is modeled as softmax of weighted sums of two vectors, decision vector at current turn, y_s^t and previous belief state, b_{t-1}^s.

$$b_s^t = softmax(W_{curr} y_s^t + W_{prev} b_s^{t-1}) \tag{3.1}$$

This update mechanism learns two matrices, W_{curr} and W_{prev}, which combine two aforementioned vectors to decide the current belief state for the slot. However, this update process limits NBT model's scope to slot-values which are encountered during the training process. Second variant named *Constrained Markovian Update*, which is designed to overcome this limitation, utilizes the fact that the new belief state of a particular slot is either equal to previous state or the state mentioned in the current utterance. *Parameter tying* mechanism proposed with the second variant preserves NBT's ability to handle unseen values. [44] extends the NBT model to a teacher-student framework which enables *cross-lingual transfer learning* from source language (teacher) to target language (student). Proposed framework focuses on zero-shot DST with no annotated dialog corpus available for students. Two scenarios:

- bilingual corpus is available between source and the target languages
- only bilingual dictionaries are available

are considered separately and propose two different knowledge transferring methods.

3.2.3 Encoder Modification with Self-Attention

GLAD [45] addresses the challenge of detecting slot-values with very few training examples. This is decisive because chances that given utterance contain one of many rare slot-values is significant even though, probability of encountering a given rare slot-value is low. In a high-level architectural perspective, both NBT and GLAD are quite similar despite significant differences in module names used by authors. Novelty of GLAD comes from global-locally self-attentive encoder which is used within all three encoders (Action encoder, Utterance Encoder, Slot-value encoder) internally. In contrast to most the DST models which predicted each slot-value pair in isolation ([46] discussed earlier being an exception), GLAD encoder introduces global modules to share parameters among slot-value pairs along with local modules which learn slot-specific features.

Given an input X, proposed encoder outputs encoding (H) and the self-attention context (c). Encoder mapping of GLAD can be denoted as:

$$encode(X) -> H, c \tag{3.2}$$

Latency due to recurrent networks is a major limitation of GLAD. For each utterance, GLAD needs to evaluate all slot-value pairs, with a single slot-value at time, which significantly increase the computational time of the model. This makes GLAD inefficient in production settings. Addressing these issues, [25] propose Globally-Conditioned Encoder (GCE), improving GLAD encoder architecture removes slot-specific recurrent and self-attention layer by concatenating utterance encoding and action encoding with slot embedding vectors. Modified encoding function can be denoted as:

$$encode(X, s) -> H, c \tag{3.3}$$

[47] introduce the 'Candidate Set' concept for improved scalability in multi-domain DST. Intuition is that a slot-value that has not been mentioned in dialogue up to that turn has near zero probability of being the belief state. Therefore, the probability distribution over the selected *candidate set* is calculated. Imposing an upper bound(K) on the size of the *candidate set* allows deployed dialogue systems to introduce new slot-values without affecting the computational time of the system. Paper proposes a candidate scorer mechanism to extract top-K candidates and final distribution is over $K + 2$ candidates with *none* and *dontcare* conditions always considered. Related to feature selection, three types of features are defined. Utterance related features are relevant to all the candidates under all the slots. Scope of slot related features is limited to all the candidates under a particular slot. Candidate related features are relevant only to a specific candidate. Same, two layered, bi-directional GRU network, which takes original utterance and the delexicalized utterance (delexicalizing values only) to respective layers, is used to extract:

- utterance representation
- candidate level details from user and system utterances

Extraction from last user and system utterances are concatenated with last dialogue and system acts to define respective types of features. Slot related features are defined in terms of system and user acts, which involve the slot being considered (such as *request(s), deny (s)*) and score estimates for none and dontcare conditions. Dimensions of candidate scoring mechanism related model-parameters are independent of slots while dimension of two layered, bi-directional GRU network being independent of domain. These characteristics allow model-parameters to be transferred from one domain to another. This cross-domain transferability is a vital contribution for the progression of research in DST.

3.2.4 Reading Comprehension Approach with Pointer Networks

Dialogue systems are usually designed with predefined ontology and most the low-resource adaptation research discussed previously targeted either on rare slot-values (very few examples in training-set) or unseen values (no occurrence in training-set). However, both of these scenarios expect possible slot-values to be predefined in the domain ontology.

[48] propose a solution to the relatively unaddressed problem of detecting unknown (slot-value not defined in the ontology) slot-values in state tracking. Proposed model is based on a pointer network (PtrNet) architecture [28] which outputs discrete tokens corresponding to positions in an input sequence. Two main differences of PtrNet from standard seq2seq models are:

- output is discrete and corresponds to a position in input sequence
- number of target classes vary depending on the size of the input

Drawing parallel with machine comprehension, problem of DST is reformulated as a pointing task, which output of the decoder indicates the starting and the ending indices of the input sequence. Model embeds entire dialogue history up to current turn t, while augmenting with speaker role to differentiate between user and system utterances. Bidirectional LSTM encoder is used for encoding. Even though, the proposed techniques succeed in pointing to the value when it is mentioned, it fails to identify none and dontcare conditions, because there is no part of the sequence to be pointed. Therefore, a classification component is introduced to overcome the problem of *non-pointable* values, classifying *pointable, none* and *dontcare.* Final forward state of the encoder is used as an input feature for the classifier. Identifying, insufficient tail representation with the same set of slot values repeating frequently as a major cause for poor generalizability, targeted dropout mechanism, is introduced to improve performance. The solution is an adaptation of widely used randomize dropout [29]. However, in the targeted dropouts, frequent slot-value embeddings are selectively set to zero. This forces the model to learn unknown slot-values from the context rather than remembering frequently occurring words.

[1] very explicitly takes a reading comprehension approach, with targeted questions being *"what is the value of slot i ?"* with i being the name of the slot . For a dialogue system which tracks M types of slots, M number of questions will be asked. Attention based networks which find the span of the answer to the aforementioned question make the core of the model. This approach overcome the necessity for fixed ontology and enable tracking unknown slots. Prior to the span component, two components are added to the model in the following order: First, slot carryover model which makes a binary decision on whether the slot-value from the previous turn to be used. Second, a classifier which classify answers among target classes: *yes, none, dontcare, span*. This classification process takes place only if the carryover model decided not to carryover. *Yes* refers to categorical states (such as requestable slots) while span refers to named entities within the dialogue. The slot span model will be used to extract entities only when classified under span. Successful use of pre-trained contextual word embedding [49] is another noteworthy contribution of this work.

[50] makes a significant contribution for the direction of multi-domain low-resource research. Model is made of three components, Utterance Encoder, State Generator and Slot Gate *Utterance Encoder*: Authors use bidirectional GRU for encoding. However, the model does not make any strict limitation on the encoding model. Last l dialogue utterances(U_i) and system responses(R_i) sequence is taken as inputs rather than last utterance. Where d_{emb} refers to the size of word embedding:

$$X_t = (U_{t-l}, R_{t-l}, \dots . U_t, R_t) R^{|x_t| * d_{emb}} \tag{3.4}$$

State Generator: Similar to [27], TRADE also utilizes a pointer-network approach in state generators. However, deviating from the index-based pointer network, TRADE uses a soft-gated pointer-generator model. This model can be thought of as a hybrid of sequence to sequence model with PtrNet. While PtrNet provide a copying mechanism from the input, attention based seq2seq gives generative capabilities to the model. These hybrid characteristics are preferred over index based PtrNet as there is no guarantee that exact words of true slot-values are present in the text [1]. GRU decoder value for each (domain, slot) pair is predicted by the generator. Summed embedding of the domain and slot are given as first inputs followed encoder input. Resulted decoder output, utilized for two purposes.

- compute history attention
- as an input of final output distribution calculation after mapping to vocabulary space.

Slot Gate: For all the (domain, slot) pairs, this three-way classifier takes encoder hidden state as input feature and classifies among *ptr, none and dontcare*. Decoder output is ignored when the slot gate predicts none or dontcare.

Paper empirically shows its domain-adaptation capabilities in zero-shot and few-shot settings. Zero-shot adaptability is demonstrated by excluding one domain at a time and training on remaining domains. Excluded domain is used for testing. Domain expandability of multi-domain dialogues is an important yet overlooked aspect of DST research. In the case of domain adaptation, the model is further trained to learn a new domain with performance of the originally trained domain not being a concern. However, multi-domain dialogues expect models to perform well on expanded domains without forgetting (or unlearning) previously trained domains. TRADE is evaluated for domain expandability with promising results in adapting to new domains.

[1]For more details on soft-gated pointer generator, user is referred to [51]

Chapter 4

METHODOLOGY

Meta-learning work in the task-space compared to the data-space which normal supervised learning work. Each individual task here contains set of utterances with its intent and slot-tags. For $n - way\ k - shot$ joint intent detection, a task should be accompanied by total of $n * k$ utterances with k utterances under each of n intents. D_{mtr}, meta-dataset is made of such datasets under different tasks. Further we maintain k_{mevl} sample utterances under each n intents for meta-evaluation. These meta-evaluation datasets combines to form D_{mevl} meta-dataset. Structure of adaptation phase meta-datasets D_{ft} and D_{eval} are similar to D_{mtr} and D_{mevl} respectively. We formulate the problem assuming the existence of pool of tasks with datasets having aforementioned characteristics. Details on how these pools of tasks are generated for each experiment is discussed in section 5.1.1.

4.1 Meta Learning Problem Formulation

Pool of meta-learning tasks τ_{mtl} can be formally summarized as:

$$\tau_{mtl} \ = \left\{\tau^1_{mtl},\ \tau^2_{mtl},\ ..\ ,\ \tau^L_{mtl}\right\} \tag{4.1}$$

Each of these tasks τ^i_{mtl} is accompanied with a meta-training D^i_{mtr} and meta-evaluation set D^i_{mevl}. With $(x^i_a, y^i_a\ )$ indicating input utterance and target label pair, we can define:

$$\begin{aligned} D_{mtl} &= \left\{(D^1_{mtr}, D^1_{mevl}), ..., (D^L_{mtr}, D^L_{mevl})\right\} \\ D^i_{mtr} &= \left\{(x^i_1, y^i_1), ...(x^i_m, y^i_m)\right\} \end{aligned} \tag{4.2}$$

Adaptation tasks τ_{adp} can be defined in a similar manner. If D^{new}_{ft} and D^{new}_{evl} are fine-tuning and evaluation datasets of task τ^{new}_{adp} entire learning process can be denoted as:

$$\phi^* = \arg\max_{\phi} \log \Pr \left(\phi | D_{adp}^{new}, D_{mtr}\right) \tag{4.3}$$

Learning from D_{mtr} is summarized to θ as follows:

$$\begin{aligned} &\theta : \Pr(\theta | D_{mtr}) \\ &\theta^* = \arg\max_{\theta} \log \Pr \left(\theta | D_{mtr}\right) \end{aligned} \tag{4.4}$$

Finally, with the prior θ^*, the problem can be redefined as :

$$\phi^* = \arg\max_{\phi} \log \Pr \left(\phi | D_{adp}^{new}, \theta^*\right) \tag{4.5}$$

Meta-JOSFIN algorithm proposed in Section 4.2 can be used to find this prior.

4.2 Meta JOSFIN Algorithm

Algorithm 2 Meta JOSFIN: Meta Learning for Joint Intent Detection and Slot-Filling

```
Require: $\tau_{mtl}$
Require: : $\alpha, \beta$
 1: while training do
 2:   Sample batch of tasks $A$ from $\tau_{mtl}$
 3:   for all $a$ in $A$ do
 4:     for $j$ in $N$ do
 5:       Evaluate $\nabla_\theta L_{D^a_{mtr}}(f_{\theta^a_{j-1}})$
 6:       Compute adapted parameters with gradient descent:
          $\theta^a_j := \theta^a_{j-1} - \alpha \nabla_\theta L_{D^a_{mtr}}(f_{\theta^a_{j-1}})$
 7:       Compute and store $L_{D^a_{mevl}} f_{\theta^a_j}$
 8:     end for
 9:   end for
10:   $\theta := \theta - \beta \nabla_\theta \sum_{a=1}^{A} \sum_{j=0}^{n} v_j L_{D^a_{mevl}} f_{\theta^a_N}$
11: end while
```

Given a pool of joint intent detection and slot-filling tasks, τ_{mtl}, our proposed approach finds a prior θ^* such that, model can adapt to new tasks and environments with minimum number of examples. Proposed procedure is shown in Algorithm 2. From the tasks distribution, batch of A $n - way$, $k - shot$ joint intent detection tasks are sampled . (Characteristic of these meta-learning tasks and the attached datasets is discussed in sub-section 5.1.1)

We define the joint intent detection base-model to be f_θ with meta-parameter θ. Let τ^a is a meta-learning task with meta-training set D^a_{mtr} and meta-evaluation set D^a_{mevl} where a is the index of the task. Goal is to learn an initial parameter $\theta = \theta_0$ for the f_θ such that, few N number of gradient updates using D^a_{mtr} suffice for the model to perform well on meta-evaluation set D^a_{mevl}.This N updates are known as *inner-loop update* and j^{th} inner loop update can be denoted as:

$$\theta^a_j := \theta^a_{j-1} - \alpha \nabla_\theta L_{D^a_{mtr}}(f_{\theta^a_{j-1}}) \tag{4.6}$$

α is the task learning rate and θ^a_j is the base-model weights after j^{th} inner loop update. $\nabla_\theta L_{D^a_{mtr}}(f_{\theta^a_{j-1}})$ represent the loss on support set after $j-1$ updates.

Following N inner loop updates on the batch B, we can define a *meta-objective* as the sum of base-model losses on meta-evaluation set with initialization θ^a_N. With $\theta^a_W(\theta_0)$ explicitly denoting the dependence of θ^a_N on θ_0 meta-objective:

$$L_{meta}(\theta_0) = \sum_{a=1}^{A} L_{D^a_{mevl}} f_{\theta^a_W(\theta_0)} \tag{4.7}$$

Meta-objective 4.2 indicates, how good is our initialization with respect to a given task. Now the meta objective can be optimized with outer-loop updates:

$$\theta_0 := \theta_0 - \beta \nabla_\theta \sum_{a=1}^{A} L_{D^a_{mevl}} f_{\theta^a_N} \tag{4.8}$$

[52] has shown that *Multi Step Loss Optimization*(MSL) stabilize the training over completing all the inner-loops before target set loss evaluation. MSL calculate the total loss as weighted sum of losses on D^a_{mevl} after every inner-loop step.When v_i is the importance weight of the target set loss at step i, outer-loop update changes to:

$$\theta_0 := \theta_0 - \beta \nabla_\theta \sum_{a=1}^{A} \sum_{j=0}^{n} v_j L_{T^a} f_{\theta^a_N} \tag{4.9}$$

Adaptation phase is similar to meta-learning phase with the key difference of omitting outer-loop update. We initialize the model with learnt θ^* and fine tune for specific tasks, performing N inner-loop updates.

4.3 Base-Model Changes

Deviating from calculating loss by simple arithmetic sum of two goals in base-model, we introduce hyper-parameter η to our base-model as of Equation :4.3, to control learning process of intent detection and slot-filling. This η can be adjusted according to the distribution of slot tags over intents.

$$Loss = \eta Loss_i + (1 - \eta) Loss_{sf} \tag{4.10}$$

Chapter 5

EVALUATION

5.1 Experimentation

The goal of our experiment and evaluation is to answer the following question: Can our meta-learning approach learn an initial parameter θ_0 such that, the model can adapt to similar joint intent detection task within few-shot setting. We evaluate the applicability of our meta-learning approach in two different few-shot classification settings.

- meta-learnt with one set of intents, evaluate the adaptability for new set of intents
- meta-learnt with one language, evaluate the adaptability for new languages

5.1.1 Task Pool Generation

As mentioned before, meta-learning learns in the task-space. Therefore, how we generate our pool of tasks from available data is an important research problem. To generate a pool of tasks we need a dataset of datasets. First, we assume the existence of a single dataset with following characteristics:

- There exist a separate train-set and test-set with labelled utterances under set of intents.
- Slot-tags across these intents are consistent. For example, if the tag *'time'* is used to denote a specific time of a day in one intent, it should be used to denote the same property over all intents.

Train set in the dataset is further split into two where one split is used for generating meta-train and the other is sampled for meta-evaluation utterances. From set of intents available, we randomly select n intents. Then we sample k

labelled examples from each intent creating $n - way$, $k - shot$ meta-train tasks. For the same set of intents, we sample k_t utterances for each intent, from the other split. We follow the same procedure with test-set to generate fine tuning and evaluation tasks.

5.1.2 Datasets

In order to test adaptability for unseen intents, we utilized ATIS dataset. Original ATIS training-set consist of 4978 utterances over 17 intents. We selected 9 intents for generating meta-learning tasks while utterances under remaining 8 intents were used for evaluating the adaptability for unseen intents. However, our hypothesis demands the number of examples available even for training (meta-learning) task to be limited. To simulate this constraint, we reduced the original ATIS dataset such that maximum number of utterances per intent in reduced meta-learning dataset is 30. Table 5.1 shows the details of restricted dataset. Train-set made of 9 intents were further split into two. Set of k utterances required to generate meta-training tasks are drawn from one split while the other split is used for sampling k_t utterances for meta-evaluation. This splitting ensures meta-evaluation examples differ from meta-training examples. We followed the same procedure with test-set to generate fine-tuning and evaluation dataset.

Dataset	Intents	Utterances	Constraint(per intent)	Stage Used
Train	9	112	=< 20	Meta-Training
		47	=<10	Meta-Evaluation
Test	8	Dynamic	No upper limit	Adaptation
			No upper limit	(Adaptation) Testing

Table 5.1: Reduced ATIS dataset constraints and their usage in task pool generation. Original ATIS training-set contain 4978 utterances across 17 intents. In contrast, our reduced dataset utilize 159 utterances over 9 intents for meta-learning.

For evaluating the adaptability for new languages, we used publicly available [53] dataset [1] which contains utterances in English, Spanish and Thai. Utterances are under three domains alarm, reminder and weather. In our experiment, we do not follow this domain-intent hierarchy in the given dataset and create a flat pool

[1] `https://fb.me/multilingual_task_oriented_data`

of 11 intents. Tasks generated from English language utterances across 11 intents are used during meta-learning phase. Similar to reducing ATIS dataset, we created smaller dataset from available utterances. In contrast to 30521 training examples across 12 intents in the original dataset, our reduced dataset for generating meta-training tasks contained 11 intents with 30 utterances each. To draw utterances for meta-evaluation, reduced dataset contained separate split with maximum of 30 utterances per intent. For fine-tuning and testing stages, we used Spanish and Thai datasets.

Adhering to the process described in section 5.1.1, we generated the pool of tasks required for each experiment using above reduced datasets.

5.1.3 Experimental Setup

Evaluating the adaptability in our approach for new tasks, we ran tests with 5-way 1-shot and 5-way 2-shot settings. For each setting, we generated training tasks dynamically while maintaining 50 evaluation tasks. Out of 50 evaluation tasks generated, 10 tasks used for validating after each epoch in meta-training. From these validation statistics, we chose best 5 models and tested their adaptability on 50 evaluation tasks. To compare the quality of the meta parameter learnt by our model, we ran the base-model with random initialization on same set of evaluation tasks. Pre-trained Glove [54] word embedding is used for initializing word vectors.

For evaluating adaptability to new languages 5-way 1-shot and 5-way 2-shot settings were evaluated for adaptability to new languages. We generated meta-learning tasks dynamically from English dataset and maintained the same 60 evaluation tasks from each language across all experiments. Out of 60 tasks, 12 tasks were used as validation set for evaluating the model after each epoch. For each language we trained models with

- 64 dimension randomly initialized embedding
- 300 dimension (further) trainable MUSE embedding (TE)
- 300 dimension non trainable MUSE embedding (NTE)

5.2 Results

Model	Intent Detection		Slot-Filling	
	Direct	JOSFIN	Direct	JOSFIN
5-way 1-shot	27.6	**30.4**	**35.65**	32.73
5-way 2-shot	31.2	**40.2**	23.7	**38.6**

Table 5.2: Evaluation results of adaptability for new intents on reduced ATIS dataset. Mean accuracy values in intent detection and slot-filling over 50 evaluations tasks presented.

Table 5.2 shows the results of few-shot experiments on reduced ATIS dataset. 5-way 1-shot setting shows 2.8% gain with a mean intent accuracy of 30.4%. However, slot-filling accuracy of the model with the best intent accuracy is lower compared to base-model random initialization. 5-way 2-shot intent detection task shows 9% improvement with Meta JOSFIN to achieve 40.2% accuracy. This suggests that proposed JOSFIN algorithm learns a parameter that enables rapid inferring with few examples of new set of intents. Further, it is noteworthy that 2-shot setting has shown significant improvement over 1-shot setting.

Model			Intent Detection		Slot-Filling	
			Direct	JOSFIN	Direct	JOSFIN
Spanish	5-way 1-shot	W/O Muse	20.8	39.2	22.5	**55.9**
		With Muse(TE)	30.0	34.8	33.9	55.1
		With Muse(NTE)	25.2	**43.2**	38.6	50.4
	5-way 2-shot	W/O Muse	22.2	44.8	21.9	58.6
		With Muse(TE)	36.6	**48.4**	43.1	**69.1**
		With Muse(NTE)	34.4	42.8	50.2	67.9
Thai	5-way 1-shot	W/O Muse	21.6	36.8	20.5	55.1
		With Muse(TE)	33.6	**42.4**	55.1	64.6
		With Muse(NTE)	28.0	36.8	55.3	**68.2**
	5-way 2-shot	W/O Muse	22.2	50.8	22.9	73.2
		With Muse(TE)	39.6	49.0	63.8	74.3
		With Muse(NTE)	44.8	**55.2**	67.9	**78.1**

Table 5.3: Intent Detection and Slot-Filling accuracy on Spanish and Thai datasets. *Direct* indicates random initialization of the base-model and *JOSFIN* stands for learning a prior from Meta-JOSFIN algorithm. TE: Trainable Embedding NTE: Non Trainable Embedding

Table 5.3 shows results for cross lingual adaptability experiments. In 5-way 1-shot experiment on Spanish dataset, our approach report maximum of 43.2% accuracy on intent detection with non-trainable Muse embedding where highest accuracy achieved by the base-model without Meat-JOSFIN is 30.0%. Slot-filling accuracy also improves in all 3 variation of Spanish 5-way 1-shot experiments. Similar behaviour can be seen in Thai 5-way 1-shot setting with the only exception of intent detection with trainable Muse embedding reducing the accuracy. For 5-way 2-shot experiments, trainable Muse with Meta-JOSFIN shows the best accuracy on Spanish data with 48.4% intent detection and 69.1% slot-filling accuracy. Our method achieve 55.2% intent detection and 78.1% (with non trainable Muse embeddings) in slot-filling on Thai 5-way 2-shot experiments. Further, there is a clear gain from 1-shot setting to 2-shot setting.

5.3 Discussion

Results from both experiments support our hypothesis that Meta-JOSFIN algorithm learns a meta-parameter (prior) which stores common charectersitics from similar, but different tasks. This enable rapid inference on new tasks similar to human brain.

In the reduced ATIS experiment, our approach has clearly performed better compared to random initialization approach in both intent detection and slot-filling tasks. This suggests prior found using one set of intents contains information of underlying tasks (n-way k-shot classification and slot-filling) rather than just remembering few tags. This 'prior knowledge' acquired by the meta-parameter enable fast adaptation for completely new set of intents.

In cross lingual experiment, our meta-learning based approach shows significant improvement in accuracy with only one exception (5-way 1-shot trainable Muse on Thai data). This mean our Meta-JOSFIN algorithm has been able to learn initializing parameter which stores language-independent knowledge. This language independent learning from English utterances enable the base-model to quickly adapt new languages: Spanish and Thai.

Further, it can be noted in both the experiments that there is a significant improvement from 1-shot to 2-shots. This suggests that parameter learnt by Meta-JOSFIN does not restrict the further learnability with more examples.

Transfer Learning Approaches vs Meta-JOSFIN: Usually transfer learning approaches in NLP which focus on generalizability and adaptability expects one large source task for initial training . However, this large dataset is not available in real-world scenarios. Reduced datasets used to produce meta-training and meta-evaluation datasets in our experiments are magnitudes smaller compared to source task dataset expected for transfer learning. However, in a broader sense, both meta-learning and transfer learning can be applied for low-resource settings. For example, in the cross-lingual experiment we utilized pre-trained word embedding which is a form of transfer learning.

Stabilizing Model Training: It is noted during model training that the generalization ability of the meta-parameter θ does not improve smoothly with the number of epochs. Even though there is overall upward trend against the number of epochs, five best models selected were not concentrated towards last epochs. This suggests that model training for generalizability is unstable to an extent and further investigation may be conducted.

In summary, the proposed meta-learning approach, Meta-JOSFIN encode underlying similarity from different tasks and improve the adaptability and generalizability for new tasks. However, the training process of the proposed approach is unstable to some extent and further research is required to stable training-setup.

Chapter 6

CONCLUSION

The first phase of our work targeted towards DST models with separate NLU component. Modeling the problem of intent detection and slot-filling as a few shot meta-learning problem, we empirically demonstrated that NLU module in goal-oriented dialogues can greatly benefit from learn to learn paradigm in low-resource settings. In the second phase, presented systematic study of the evolution of joint NLU/DST models with respect to generalizability and adaptability.

6.1 Future Directions

We believe, our work especially related to meta-learning lay foundation to two main line of research:

- Despite not being straight forward, adapting leave-one-out cross validation proposed in [55] is a probable improvement to the meta-learning setup.
- As described in the review of joint NLU/DST models heavily rely on transfer learning techniques for generalizability and adaptability. We view meta-learning techniques for joint NLU/DST models as a promising line of research.
- We also believe all conversational AI related tasks including DPL and NLG should shift the focus from data-driven approaches to task-driven approaches, Shallow and distributed nature of dialogue datasets would complement this line of research.

References

[1] Jianfeng Gao, Michel Galley, and Lihong Li. *Neural Approaches to Conversational AI.* now Publishers Inc, 2019.

[2] Antoine Raux, Brian Langner, Dan Bohus, Alan W Black, and Maxine Eskenazi. Let's go public! taking a spoken dialog system to the real world. In *9th European Conference on Speech Communication and Technology, Lisbon, Portugal*, September 2005.

[3] Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau. A survey of available corpora for building data-driven dialogue systems. *CoRR*, abs/1512.05742, 2015.

[4] Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. The dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, pages 404–413, Metz, France, August 2013. Association for Computational Linguistics.

[5] Matthew Henderson, Blaise Thomson, and Jason D.Williams. Handbook for the dialog state tracking challenge 2 3, 2013.

[6] Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gasic, Lina M. Rojas Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 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*. Association for Computational Linguistics, 2017.

[7] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 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. Association for Computational Linguistics, 2018.

[8] Sinno J. Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Trans. on Knowl. and Data Eng.*, 22(10):1345–1359, October 2010.

[9] Jurgen Schmidhuber. Evolutionary principles in self-referential learning. on learning now to learn: The meta-meta-meta...-hook. Diploma thesis, Technische Universitat Munchen, Germany, 14 May 1987.

[10] Yoshua Bengio and Samy Bengio. Learning a synaptic learning rule. Technical Report 751, Département d'Informatique et de Recherche Opérationelle, Université de Montréal, Montreal, Canada, 1990.

[11] Brenden Lake, Ruslan Salakhutdinov, and Joshua Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350:1332–1338, 12 2015.

[12] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy P. Lillicrap. Meta-learning with memory-augmented neural networks. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *ICML*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1842–1850. JMLR.org, 2016.

[13] Danilo Jimenez Rezende, Shakir Mohamed, Ivo Danihelka, Karol Gregor, and Daan Wierstra. One-shot generalization in deep generative models. *CoRR*, abs/1603.05106, 2016.

[14] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic Meta-learning for Fast Adaptation of Deep Networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 1126–1135. JMLR.org, 2017.

[15] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A Simple Neural Attentive Meta-Learner. In *International Conference on Learning Representations*, 2018.

[16] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *NIPS*, pages 4077–4087, 2017.

[17] Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *CoRR*, abs/1611.02779, 2016.

[18] Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[19] Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. Investigating meta-learning algorithms for low-resource natural language understanding tasks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *EMNLP/IJCNLP (1)*, pages 1192–1197. Association for Computational Linguistics, 2019.

[20] Fei Mi, Minlie Huang, Jiyong Zhang, and Boi Faltings. Meta-learning for low-resource natural language generation in task-oriented dialogue systems. In Sarit Kraus, editor, *IJCAI*, pages 3151–3157. ijcai.org, 2019.

[21] Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen tau Yih, and Xiaodong He. Natural language to structured query generation via meta-learning. In Marilyn A. Walker, Heng Ji, and Amanda Stent, editors, *NAACL-HLT (2)*, pages 732–738. Association for Computational Linguistics, 2018.

[22] Bing Liu and Ian Lane. Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling. In *Interspeech 2016*. ISCA, sep 2016.

[23] P. Haffner, G. Tur, and J.H. Wright. Optimizing SVMs for complex call classification. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP 03).* IEEE.

[24] R. E. Schapire and Y. Singer. BoosTexter: A Boosting-based System for Text Categorization. *Machine Learning*, 39(2/3):135–168, 2000.

[25] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *NIPS*, pages 932–938. MIT Press, 2000.

[26] Joo-Kyung Kim, Gökhan Tür, Asli Çelikyilmaz, Bin Cao, and Ye-Yi Wang. Intent detection using semantically enriched word embeddings. In *SLT*, pages 414–419. IEEE, 2016.

[27] Yoon Kim. Convolutional neural networks for sentence classification. 2014. cite arxiv:1408.5882Comment: To appear in EMNLP 2014.

[28] A. Bhargava, Asli Çelikyilmaz, Dilek Hakkani-Tür, and Ruhi Sarikaya. Easy contextual intent prediction and slot detection. In *ICASSP*, pages 8337–8341. IEEE, 2013.

[29] Suman V. Ravuri and Andreas Stolcke. Recurrent neural network and lstm models for lexical utterance classification. In *INTERSPEECH*, pages 135–139. ISCA, 2015.

[30] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[31] Taku Kudo and Yuji Matsumoto. Chunking with support vector machines. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*, 2001.

[32] Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. Using recurrent neural networks for slot filling in spoken language understanding. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 23(3):530–539, 2015.

[33] Shiya Ren, Huaming Wang, Dongming Yu, Yuan Li, and Zhixing Li. Joint intent detection and slot filling with rules. In Sen Hu and Lei Zou, editors, *CCKS Tasks*, volume 2242 of *CEUR Workshop Proceedings*, pages 34–40. CEUR-WS.org, 2018.

[34] Dilek Hakkani-Tür, Gokhan Tur, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM. In *Interspeech 2016*. ISCA, sep 2016.

[35] Michel Galley, Mark Hopkins, Kevin Knight, and Daniel Marcu. What's in a translation rule? In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 273–280, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics.

[36] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.

[37] Dong Wang and Thomas Fang Zheng. Transfer learning for speech and language processing. In *APSIPA*, pages 1225–1237. IEEE, 2015.

[38] Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. Transfer learning for low-resource neural machine translation. *CoRR*, abs/1604.02201, 2016.

[39] Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. Cross-lingual transfer learning for multilingual task oriented dialog. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *NAACL-HLT (1)*, pages 3795–3805. Association for Computational Linguistics, 2019.

[40] Matthew Henderson, Blaise Thomson, and Steve Young. 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)*. Association for Computational Linguistics, 2014.

[41] Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2016.

[42] Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017.

[43] Nikola Mrkšić and Ivan Vulić. Fully statistical neural belief tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, 2018.

[44] Wenhu Chen, Jianshu Chen, Yu Su, Xin Wang, Dong Yu, Xifeng Yan, and William Yang Wang. XL-NBT: A cross-lingual neural belief tracking framework. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

[45] Victor Zhong, Caiming Xiong, and Richard Socher. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2018.

[46] Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 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 (Volume 2: Short Papers)*. Association for Computational Linguistics, 2015.

[47] Abhinav Rastogi, Dilek Hakkani-Tur, and Larry Heck. Scalable multi-domain dialogue state tracking. In *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, dec 2017.

[48] Puyang Xu and Qi Hu. An end-to-end approach for handling unknown slot values in dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2018.

[49] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. In *Proceedings of the 2019 Conference of the North*. Association for Computational Linguistics, 2019.

[50] Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2019.

[51] Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2017.

[52] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your MAML. In *International Conference on Learning Representations*, 2019.

[53] Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. Cross-lingual transfer learning for multilingual task oriented dialog. *CoRR*, abs/1810.13327, 2018.

[54] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[55] Davis Wertheimer and Bharath Hariharan. Few-shot learning with localization in realistic settings. *CoRR*, abs/1904.08502, 2019.