上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

学士学位论文

THESIS OF BACHELOR

论文题目： Dialogue State Tracking in Statistical Dialogue Management

学生姓名： 孙锴

学生学号： 5110309061

专 业： 计算机科学与技术

指导教师： 俞凯

学院(系)： 致远学院

Dialogue State Tracking in Statistical Dialogue Management

ABSTRACT

Dialogue management is the core of a dialogue system. In recent years, there is a research trend towards statistical dialogue management. As an important part of statistical dialogue management, dialogue state tracking (DST) is a process to estimate the distribution of the dialogue states at each dialogue turn given the interaction history. Recently, to advance the research of statistical dialogue management, researchers start to formulate dialogue state tracking as an independent problem so that a bunch of machine learning algorithms can be investigated.

Both rule-based and statistical approaches have been successfully used for the DST problem. Since the DST problem is raised out of the statistical dialogue management framework, statistical approaches have been the natural focus and achieved the state-of-the-art performance. However, statistical approaches have also shown large variation in performance and poor generalization ability due to the lack of data. There also have been attempts of using probability operation rules for DST, due to their simplicity, efficiency and portability. However, the performance of these methods are usually not competitive to statistical tracking approaches and there lacks a way to improve the DST performance when training data are available.

In this paper, two novel frameworks are proposed which manage to take advantage of both rule-based and statistical approaches. One framework, referred to as *constrained Markov Bayesian polynomial* (CMBP), taking the first step towards bridging the gap between rule-based and statistical approaches for DST, formulates rule-based DST in a general way and allow data-driven rules to be generated. Here, a DST rule is defined as polynomial function of a set of probabilities satisfying certain linear constraints. Prior knowledge is encoded in these constraints. Under reasonable assumptions, CMBP optimization can be converted to a constrained linear programming problem. Another framework, referred to as *recurrent polynomial network* (RPN), further bridges the gap. RPN's unique structure enables the framework to have all the advantages of CMBP including efficiency, portability and interpretability. Additionally, RPN achieves more properties of statistical approaches than CMBP. CMBP and RPN were evaluated on the data corpora of the second and the third Dialog State Tracking Challenge (DSTC-2/3). Experiments showed that both CMBP and RPN have good generalization ability and can significantly outperform both traditional rule-based approaches and statistical approaches with similar feature set. Compared with the state-of-the-art statistical DST approaches with a lot richer features, CMBP and RPN are also competitive.

Keywords: Statistical Dialogue Management, Dialogue State Tracking, Rule-based Model, Statistical Model, Data-driven Rule, Constrained Markov Bayesian Polynomial, Recurrent Polynomial Network

Contents

Chapter 1 Introduction

1.1 Spoken Dialogue System

1.1.1 Spoken Dialogue System

A task-oriented spoken dialogue system (SDS) is a system that can continuously interact with human to accomplish a predefined task through *speech*. It usually consists of three modules: input, output and control, as shown in figure 1.1. The input module mainly consists of automatic speech recognition (ASR) and spoken language understanding (SLU), with which semantics-level user dialogue acts are extracted from acoustic speech signals. With the input user dialogue acts, the control module, also called dialogue management accomplish two missions. One is to maintain its internal state, an encoding of the machine's understanding about the conversation. As information is received from the input module, the state is updated, which is called *dialogue state tracking* (DST). Another mission is to choose a machine action based on its *policy*, also at semantics-level, to direct the dialogue given the information of the dialogue state, referred to as *dialogue decision making*. The output consists of natural language generation (NLG) and text-to-speech (TTS) synthesis, with which machine dialogue acts are converted to audio.

1.1.2 Dialogue Management

Dialogue management is the core of a dialogue system. Traditionally, most commercial spoken dialogue systems assume observable dialogue states and employ hand-crafted rules for dialogue management, such as dialogue flow-chart. Here, dialogue state is observable, hence no tracking is needed. Dialogue decision is simply a set of mapping rules from state to machine action. This is referred to as rule-based dialogue management. However, unpredictable user behaviour, inevitable automatic speech recognition and spoken language understanding errors make it difficult to maintain the true dia-

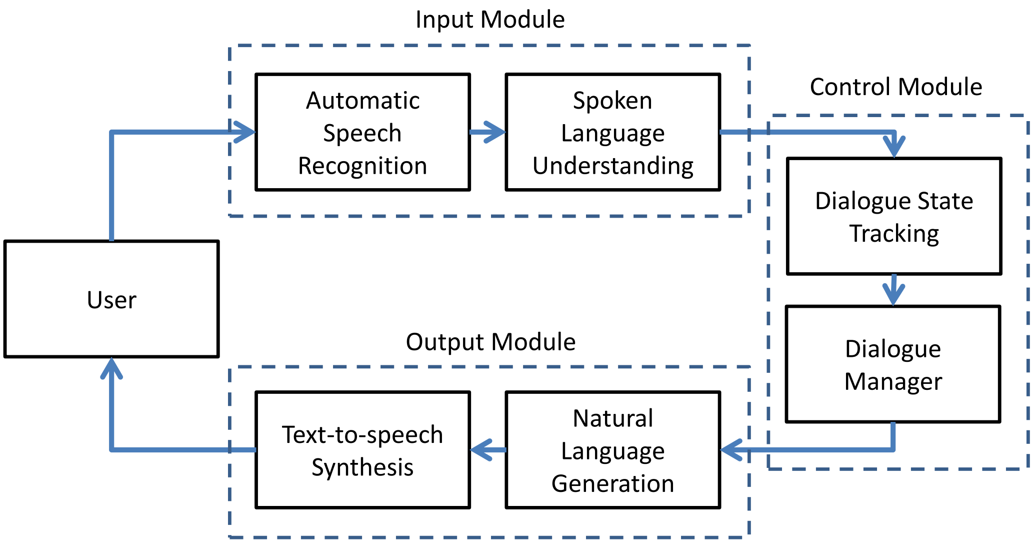

Figure 1.1 Diagram of a spoken dialogue system (SDS)

logue state and make decision. Hence, in recent years, there is a research trend towards statistical dialogue management. A well-founded theory for this is the partially observable Markov decision process (POMDP) framework ([1], Williams and Young, 2007: 393–422.)([2], Thomson and Young, 2010: 562–588.)([3], Young et al., 2010: 150–174.)([4], Young et al., 2013: 1160–1179.). In most studies of POMDP, both dialogue state tracking and decision making are modelled using statistical approaches.

Recently, to advance the research of statistical dialogue management, researchers start to formulate dialogue state tracking as an independent problem so that a bunch of machine learning algorithms can be investigated. The dialog state tracking challenge (DSTC) provides the first common testbed in a standard format, along with a suite of evaluation metrics for this purpose ([5], Williams, 2012: 959–970.)([6], Williams, 2012: 23–24.)([7], Williams et al., 2013: 404–413.)([8], Henderson et al., 2014: 263–272.)([9], Henderson et al., 2014: 324–329.).

1.2 Our Results and Organization of Thesis

1.2.1 Main Contribution of This Thesis

Two novel frameworks are proposed which manage to take advantage of both rule-based and statistical approaches. One framework is *Constrained Markov Bayesian Polynomial* (CMBP) ([10], Sun et al., 2014: 330–335.)([11], Yu et al., 2015: 1–10.) which takes the first step towards bridging the gap between rule-based and statistical approaches for dialogue state tracking. The other framework is *Recurrent Polynomial Network* (RPN) ([12], Sun et al., 2015: 1–22.) which further bridges the gap. Both frameworks achieve efficiency, portability, interpretability, simplicity as well as state-of-the-art state tracking performance.

1.2.2 Organization of This Thesis

The rest of the paper is organized as follows. Section 2 formulates the dialogue state tracking problem. The rule-based and statistical approaches for DST are reviewed in section 3, followed by two novel frameworks bridging the gap between rule-based and statistical approaches for DST. One novel framework, referred to as Constrained Markov Bayesian Polynomial, is discussed in section 4. Another novel framework, named Recurrent Polynomial Network, is introduced in section 5. Section 6 concludes the paper. Details of CMBP constraints formulation and derivative calculation of RPN are included in the Appendix.

Chapter 2 Dialogue State Tracking

A dialogue can be regarded as a time sequence $\{a_0, \mathbf{o}_1, \cdots, a_{t-1}, \mathbf{o}_t\}$, where a_i is the system information at i-th dialogue turn, including the system response, and $\mathbf{o}_i$ denotes all information from the user's speech at i-th turn, e.g. the output of ASR and SLU. At each turn, the system needs to estimate the probability distribution of the state given the whole dialogue history up to that turn, also referred to as belief state $b_t(\mathbf{s})$, or briefly b_t,

$$\begin{aligned} b_t(\mathbf{s}) &= \mathcal{T}(\mathbf{s}, b_0, a_0, \mathbf{o}_1, \cdots, a_{t-1}, \mathbf{o}_t) \\ &= P(\mathbf{s}|b_0, a_0, \mathbf{o}_1, \cdots, a_{t-1}, \mathbf{o}_t) \end{aligned} \tag{2-1}$$

where $\mathcal{T}(\cdot)$ denotes the tracker and b_0 is the initial belief state.

As shown in equation (2-1), the result of DST is influenced not only by the tracker $\mathcal{T}(\cdot)$ but also by the sequence $\{a_0, \mathbf{o}_1, \cdots, a_{t-1}, \mathbf{o}_t\}$ and the initial belief state b_0. In general, the initial state is assumed to be uniformly distributed and the system information can be obtained deterministically by the tracker. The information $\mathbf{o}_i$ from user's speech usually refers to the output of ASR and SLU. In this paper, the output of ASR is not used directly, so $\mathbf{o}_i$ denotes the output of SLU, which is an N-best list of semantic hypotheses.

For an end-to-end spoken dialogue system, a dialogue state tracker should be measured from at least the following three aspects:

- **Accuracy** The tracker should be as accurately as possible to estimate the system state. It has been shown that the improvement of tracking accuracy can benefit for the task completion rates in the end-to-end spoken dialogue system.

- **Efficiency** As shown in figure 1.1, the tracker is only a small component in the whole system. In order to make the system and users can converse in real time, the tracker should compute as fast as possible.

- **Generalization** In practice, it is hard to collect enough dialogues for training before an system is employed, which is often the case whenever a new domain encounters or the current domain is extended. Therefore, it is important that the tracker can work well in new domain or extended domain.

Partially observable Markov decision process (POMDP) framework provides a well-founded theory for statistical dialogue management. In most studies of POMDP, both dialogue state tracking and decision making are modelled using statistical approaches. In early works of POMDP, belief state is updated using Bayes' theorem with consideration of Markov and reasonable independence assumptions. This DST approach leads to the below update formula for b_t

$$\begin{aligned} b_t(\mathbf{s}_t) &= P(\mathbf{s}_t|\mathbf{o}_t, a_{t-1}, b_{t-1}) \\ &= k \cdot P(\mathbf{o}_t|\mathbf{s}_t, a_{t-1}) \sum_{\mathbf{s}_{t-1} \in S} P(\mathbf{s}_t|\mathbf{s}_{t-1}, a_{t-1}) b_{t-1}(\mathbf{s}_{t-1}) \end{aligned} \tag{2-2}$$

where k is the normalization constant and a_{t-1} is the system response in the $(t-1)$-th turn. Due to the huge number of possible states, approximation is necessary for DST in real world SDS tasks. State space partition (hidden information state, i.e. HIS) ([3], Young et al., 2010: 150–174.) or further state independence assumption (Bayesian network update state, i.e. BUS) ([2], Thomson and Young, 2010: 562–588.) have been used. However, these generative methods can neither accurately nor efficiently track the dialogue state.

To advance the statistical dialogue management research, the *Dialog State Tracking Challenge* (DSTC) is organized to provide common testbeds for comparing different DST models. There have been 3 challenges, each with a different task. All challenges have the task of tracking users' goals and employ labelled dialogue corpus and simplified dialogue state representations. The data for DSTC-1 ([7], Williams et al., 2013: 404–413.) was collected using the Let's Go system, which provides bus schedule information in Pittsburgh, USA. In this domain, there are 9 slots, such as *route, from.desc, to.desc* etc. The user can inform the system the value of any slot. As the dialogue progresses, the values for different slots are accumulated to form the user's goal. A number of different evaluation metrics were investigated in DSTC-1. Accuracy of joint goals and Brier score were accepted later as the primary metrics for future challenges.

In DSTC-2 ([8], Henderson et al., 2014: 263–272.), the domain is changed to restaurant search with 8 slots. Some slots are requestable slots, while the others are informable, which may be provided as search constraints. Dialogue state also becomes richer. In addition to the users' goals, search method (the way the user is trying to interact with the system, e.g. *by-name, by-constraints* etc.) and requested slots are also employed. DSTC-3 ([9], Henderson et al., 2014: 324–329.) focuses on dialogue domain extension. Only a small set of labelled dialogues in a new domain (tourist information) are available and all participants are asked to build a belief state tracker on the small data set plus the DSTC-2 data. The new domain has 13 slots, which include all slots in DSTC-2 and 5 new slots. In all the 3 challenges, the dialogue state tracker receives SLU N-best hypotheses for each user turn, each hypothesis having a set of act-slot-value tuples with a confidence score. The dialogue state tracker is supposed to output a set of distributions of the dialogue state. In this paper, the joint goal tracking, which is the most difficult and general task of DSTC-2/3, is of interest.

Chapter 3 Rule-based and Statistical Approaches for DST

In general, there are two types of approaches for dialogue state tracking – statistical approach and rule-based approach.

3.1 Statistical Models

3.1.1 Generative Statistical Models

In early work of POMDP, the belief state tracking is achieved by applying Bayesian rules as well as reasonable independence assumptions of the state components ([3], Young et al., 2010: 150–174.):

$$\begin{aligned} b_{t+1}(\mathbf{s}) = \eta\ p(\mathbf{o}_{t+1}|\mathbf{u}_{t+1})P(\mathbf{u}_{t+1}|\mathbf{g}_{t+1}, a_t) \\ \sum_{\mathbf{g}_t} P(\mathbf{g}_{t+1}|\mathbf{g}_t, a_t) \sum_{\mathbf{h}_t} P(\mathbf{h}_{t+1}|\mathbf{u}_{t+1}, \mathbf{g}_{t+1}, \mathbf{h}_t, a_t) b_t(\mathbf{g}, \mathbf{h}) \end{aligned} \tag{3-1}$$

In the above equation, there are only two pieces of external input information: $p(\mathbf{o}_{t+1}|\mathbf{u}_{t+1})$ which is usually approximated by the estimated distribution (usually normalised confidence score) of the semantic hypotheses $q(\mathbf{u}_{t+1})$ and $b_t(\mathbf{g}, \mathbf{h}) = \sum_{\mathbf{u}_t} b_t(\mathbf{s})$ of turn t. The rest are all model parameters: $\eta = p(\mathbf{o}_{t+1}|a_t, b_t(\mathbf{s}))$ is a constant independent of $\mathbf{s}_t$, $P(\mathbf{g}_{t+1}|\mathbf{g}_t, a_t)$ is the *user goal model*, $P(\mathbf{u}_{t+1}|\mathbf{g}_{t+1}, a_t)$ is the *user action model* and $P(\mathbf{h}_{t+1}|\mathbf{u}_{t+1}, \mathbf{g}_{t+1}, \mathbf{h}_t, a_t)$ is the *dialogue history model*.

The dialogue history model is usually deterministic and simply measures the consistency between the updated dialogue state and the original dialogue state (e.g. if a goal is denied, it is set as -1). Parameters of the other two models need to be estimated using training data separately on static corpora or optimised jointly together with policy using reinforcement learning. Hence the generative Bayesian belief estimator is regarded as a statistical DST model. It has also been applied to DSTC by concentrating only on the goal component $\mathbf{g}$, but did not yield competitive result due to inaccurate estimation of the parameters.

3.1.2 Discriminative Statistical Models

Before the DSTCs, most DST approaches are Bayesian generative models. Although they are mathematically sound, it is hard to incorporate rich features for DST and sometimes they are intractable ([13], Lee and Eskenazi, 2013: 414–422.). Hence, discriminative models, such as Maximum Entropy model (MaxEnt) ([13], Lee and Eskenazi, 2013: 414–422.)([14], Sun et al., 2014: 318–326.), Condition Random Field (CRF) ([15], Lee, 2013: 442–451.)([16], Kim and Banchs, 2014: 332–336.), Deep Neural Networks (DNN) ([17], Henderson et al., 2013: 467–471.)([14], Sun et al., 2014: 318–326.), Recurrent Neural Networks (RNN) ([18], Henderson et al., 2014: 292–299.)([9], Henderson et al., 2014: 324–329.) and Decision Forest ([19], Williams, 2014: 282–291.) etc., have been used as the discriminative statistical DST models and achieved great success since DSTC-1. There approaches fall into four main categories: *binary classification model*, *multi-classification model*, *structured discriminative model* and *labelling model*.

- **Binary Classification Model:** Here all slots are assumed to be *independent* of each other, leading to efficient state factorization:

$$b(s_1 = v_1, \cdots, s_n = v_n) = \prod_j b(s_j = v_j) \tag{3-2}$$

  In addition, for a slot, all candidate values which have not been observed up to the current turn are clustered together as a special value "$None$". This significantly reduces the computational cost.

  With these assumptions, the joint goal can be got by calculating the belief $b(s = v)$ for each slot s and candidate value v. This can be converted into a binary classification problem of determining whether $s = v$ is true or false. To reduce the number of binary classifiers, value v is encoded into input features. Hence, we only need to construct a binary classifier for each slot. Various models have been used within this framework, such as MaxEnt ([13], Lee and Eskenazi, 2013: 414–422.)([14], Sun et al., 2014: 318–326.) and DNN ([17], Henderson et al., 2013: 467–471.)([14], Sun et al., 2014: 318–326.).

- **Multi-Classification Model:** In the binary classification models, the belief of every candidate value v is evaluated separately, which may reduce the performance. To address this issue, multi-classifier is used to track the belief of all values simultaneously. Same as the binary classification models, different slots are assumed to be independent of each other, thus the belief state of each slot can be updated separately, and the belief of joint goal is calculated by equation (3-2).

  A typical example is the use of RNN ([18], Henderson et al., 2014: 292–299.)([9], Henderson et al., 2014: 324–329.). It is worth noting that training a multi-classification RNN requires sufficient examples for *every* candidate value, which is usually not possible. Hence, some approximations, such as incorporating previous belief for value v and $None$, must be used.

- **Structured Discriminative Model:** In both binary and multi-classification models, slots are assumed to be independent of each other. However, considering relational constrains may result in potential improvement of the DST performance. For instance, in the Let's Go domain, the arrival place and the departure place should not be the same ([15], Lee, 2013: 442–451.). *Structured discriminative models* are proposed to capture the relationship between slots at a particular turn.

  A typical example is CRF with manually designed factored graph ([15], Lee, 2013: 442–451.). Although CRF can capture the relationship between different slots, the CRF graph structure needs to be designed manually by experts. When the relational constraints are very complex, the design of the structures will be time-consuming. In order to tackle this problem, a web-style ranking model (decision forest model) is proposed to track the belief state of joint slots ([19], Williams, 2014: 282–291.). This model can automatically build conjunctions of raw features.

- **Labelling Model:** Although structured discriminative models utilize the relational constraints between different slots, they only focus on information of a single turn. It is also important to capture the relationship between multiple turns. In DSTC-2, a sequential labelling model is proposed to handle this ([16], Kim and Banchs, 2014: 332–336.). In this approach, the output of the model includes labels of the dialogue state from multiple turns. To model the temporal relationship, a linear-chain CRF was used ([16], Kim and Banchs, 2014: 332–336.).

3.1.3 Features

It is worth nothing that features play an important role in statistical approaches. In the DSTCs, the available information includes speech recognition and semantic parsing results as well as the system response history. Since N-best results are also available, various features such as confidence scores, ranks, statistics of confidence scores etc. can be used as features. For the speech recognition results, the most common feature is the n-gram feature weighted by confidence scores ([18], Henderson et al., 2014: 292–299.). The system dialogue acts also can provide useful information for state estimation ([18], Henderson et al., 2014: 292–299.)([14], Sun et al., 2014: 318–326.). Besides these features, the *turn-id* of the dialogue, whether the user has interrupted the system etc. can also be used as features ([17], Henderson et al., 2013: 467–471.)([13], Lee and Eskenazi, 2013: 414–422.)([14], Sun et al., 2014: 318–326.).

3.2 Rule-based Approaches

Since the DST problem is raised out of the statistical dialogue management framework, statistical approaches have been the natural focus. However, statistical approaches have also shown large variation in performance and poor generalisation ability due to the lack of data. There have been also an attempt to employ rule-based methods for dialogue state tracking due to its simplicity, efficiency, portability and interpretability. For example, the standard POMDP belief update, can be seen as a rule-based model, when all parameters are set according to prior knowledge without data-driven estimation ([20], Zilka et al., 2013: 452–456.). During DSTCs, a couple of rule-based models have also been proposed. For example, the baseline of DSTC just employs a simple rule of selecting the SLU hypothesis with the highest confidence score so far and discarding the rest ([7], Williams et al., 2013: 404–413.)([8], Henderson et al., 2014: 263–272.)([9], Henderson et al., 2014: 324–329.). In DSTC-1, the simple rule-based system outperformed many discriminative models and was ranked the 5^{th} in the joint goal tracking task on Test3. More complex and interesting rules have also been proposed to enhance the power of rule-based models.

In DSTC-1, Wang et al. ([21], Wang and Lemon, 2013: 423–432.) proposed a rule-based model according to basic probability calculation formula for events co-occurrence. In this approach, at the t-th turn of a dialogue, for slot s and value v, $P_t^+(v)$ and $P_t^-(v)$ are used to denote the sum of all the positive (user informing or affirming) or negative (denying or negating) confidence scores assigned by the SLU. The belief of "the value of slot s being v in the t-th turn", denoted as $b_t(v)$, can be calculated as follows:

- If $v \neq$ '$None$',

$$b_t(v) = \left(1 - (1 - b_{t-1}(v))(1 - P_t^+(v))\right)\left(1 - P_t^-(v)\right) \tag{3-3}$$

- Otherwise,

$$b_t(v) = 1 - \sum_{v' \neq \text{'}None\text{'}} b_t(v') \tag{3-4}$$

From the above formula, state tracking is very efficient. Besides, this improved rule-based model has outperformed most trackers in DSTC-1 (ranked the $5^{th}/2^{nd}/2^{nd}/6^{th}$ on Test1-4 respectively) and been used as a strong baseline for DSTC-2 and DSTC-3.

Other rule-based models have also been proposed. In Zilka et al. (2013), the generative Bayesian DST model is employed but the parameters are set according to rules. In Kadlec et al. (2014), system act is further introduced as a condition to determine rules under the Bayesian probability operation framework. These refined rule-based models have achieved very good tracking performance. However, most of them are still not competitive to data-driven statistical models. What's more, once the rule is set, they are not able to improve when more training data become available, hence lack the ability of evolution.

Chapter 4 Constrained Markov Bayesian Polynomial

4.1 Constrained Markov Bayesian Polynomial

Rule-based models ([20], Zilka et al., 2013: 452–456.)([21], Wang and Lemon, 2013: 423–432.), and Bayesian generative models ([3], Young et al., 2010: 150–174.) are all based on Bayes' theorem. Since Bayes' theorem is essentially summation and multiplication of probabilities, they can be rewritten in a general form, referred to as *Markov Bayesian Polynomial* (MBP):

$$b_{t+1}(\mathbf{s}) = \mathcal{P}(b_t(\mathbf{g},\mathbf{h}), q(\mathbf{u}_t)) \tag{4-1}$$

where $b_{t+1}(\mathbf{s})$ is the belief state of $\mathbf{s}$ at the t^{th} turn, $q(\mathbf{u}_t)$ is the estimated confidence distribution of the current user act $\mathbf{u}_t$ and $\mathcal{P}(\cdot)$ is a multivariate polynomial function

$$\mathcal{P}(x_1,\cdots,x_D) = \sum_{0\le k_1\le\cdots\le k_n\le D} w_{k_1,\cdots,k_n} \prod_{1\le i\le n} x_{k_i} \tag{4-2}$$

where $k \in \{0,1,\cdots,D\}$, D is the number of input variables, $x_0 = 1$, n is the order of the polynomial. The scalar coefficient

$$w_{k_1,\cdots,k_n} = f_{k_1,\cdots,k_n}(\mathbf{s}_t, a_t, \mathbf{s}_{t+1})$$

is the *parameter* of MBP. In general, they can be viewed as a function of the interaction history.

It can be observed that the Bayesian generative model, equation (3-1), is a special case of MBP with $D = n = 2$ and the MBP parameters correspond to the model parameters described in 3.1.1. Since these parameters can be estimated from data[1], the generative Bayesian belief estimator is usually regarded as a statistical DST model. It is worth noting that it is usually hard to get sufficient annotated data to estimated

[1] Except that the dialogue history model is usually manually set.

the parameters, hence, heuristics are usually used to directly optimise dialogue state tracking performance or the parameter update is performed together with the dialogue policy update within reinforcement learning framework ([3], Young et al., 2010: 150–174.).

Assuming that slots are independent and only goal tracking is of interest, for a specific slot, rule-based models, e.g. equation (3-3), can also be written in a similar form of MBP

$$b_{t+1}(v) = \mathcal{P}(b_t(v), P_t^+(v), P_t^-(v))) \tag{4-3}$$

In contrast to the generative Bayesian model, all coefficients in (4-3) are manually set to be integers. Therefore, rule-based model can be viewed as an MBP with features of $b_t(v), P_t^+(v), P_t^-(v)$ and prior knowledge (i.e. rule) is incorporated by manually setting the integral polynomial coefficients.

4.1.1 Generalized Rule-based Model: Constrained MBP

MBP gives a common form for rule-based and statistical generative Bayesian models. However, it does not provide a roadmap to bridge the two type of models. The key issue is how to allow rule-based model to be data-driven without losing the ability to incorporate prior knowledge. Here, a novel framework, constrained Markov Bayesian Polynomial (CMBP) is proposed to address issue. The basic idea is to construct a constrained optimisation problem for DST model training, where the model takes the form of MBP and the constraints encode all necessary probabilistic conditions as well as prior knowledge or intuition. In this paper, CMBP is derived as an extension of the rule-based model (3-3), hence slot and value independence are also assumed, though CMBP is not limited to the assumptions. To enhance the power of rule-based model, more probabilistic features are introduced into CMBP as below

- $P_t^+(v)$: sum of scores of SLU hypotheses informing or affirming value v at turn t
- $P_t^-(v)$: sum of scores of SLU hypotheses denying or negating value v at turn t

- $\tilde{P}_t^+(v) = \sum_{v' \notin \{v, \texttt{None}\}} P_t^+(v')$
- $\tilde{P}_t^-(v) = \sum_{v' \notin \{v, \texttt{None}\}} P_t^-(v')$
- b_t^r: probability of the value being 'None' (the value not mentioned) at turn t
- $b_t(v)$: belief of "the value being v at turn t"

With the above probabilistic features, a *Constrained Markov Bayesian Polynomial* (CMBP) model is defined as

$$b_{t+1}(v) = \mathcal{P}\left(P_{t+1}^+(v), P_{t+1}^-(v), \tilde{P}_{t+1}^+(v), \tilde{P}_{t+1}^-(v), b_t^r, b_t(v)\right) \\ \texttt{s.t. constraints} \tag{4-4}$$

The *constraints* in equation (4-4) can be classified into three categories.

- **Probabilistic constraints** enforce the probabilistic requirement by definition. These constraints can be directly written as a set of linear equality or inequalities. For example

$$b_t^r = 1 - \sum_{v' \neq \texttt{None}} b_t(v') \tag{4-5}$$

- **Intuition constraints** encode intuitive prior knowledge (i.e. rules). For example, the rule *"goal belief should be unchanged or positively correlated with the positive scores from SLU"* can be represented by

$$\frac{\partial \mathcal{P}(P_{t+1}^+(v), P_{t+1}^-(v), \tilde{P}_{t+1}^+(v), \tilde{P}_{t+1}^-(v), b_t^r, b_t(v))}{\partial P_{t+1}^+(v)} \geq 0 \tag{4-6}$$

- **Regularization constraints** attempt to regularise the solution to prevent overfitting in the data-driven rule generation in section 4.1.2. For example, the coefficients of $\mathcal{P}(\cdot)$ may be limited to be in $[-1, 1]$.

Although constraints can be represented in mathematic forms, to construct a feasible constrained optimisation problem, it is necessary to further approximate the constraints using linear equalities or inequalities. For example, equation (4-6) can be approximated by the below linear constraint

$$\left\{ \mathcal{P}(\mathbf{a}) \geq \mathcal{P}(\mathbf{b}) \;\middle|\; \begin{array}{l} \mathbf{a}, \mathbf{b} \in \boldsymbol{\chi}, a_i \in T_5 \\ a_1 = b_1 + 0.1, \quad a_i = b_i \quad \forall i \neq 1 \end{array} \right\} \tag{4-7}$$

where $\mathbf{a}$ and $\mathbf{b}$ are the 6-dimensional input vectors of equation (4-4), $\boldsymbol{\chi}$ denotes all possible input vectors and $T_5 = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$ is quantised interval of $[0, 1]$. Details of CMBP constraints and their corresponding linear approximations can be found in the appendix.

4.1.2 Data-driven Rule Generation for CMBP

Once rule-based model is formulated as CMBP, intuition knowledge becomes *soft* constraints and there usually exist multiple feasible solutions. It is then possible to employ data-driven criterion to optimise CMBP. In CMBP, polynomial order n, as shown in equation (4-2), determines the model complexity. Order $n = 1$ or $n = 2$ is too small to model complex situations, while $n \geq 4$ is too large to efficiently optimise. Hence, in this paper, polynomial order $n = 3$ is used to construct the search space. By using the overall goal tracking accuracy on the training data as the optimisation criterion, the data-driven CMBP can be written as the below optimisation problem

$$\begin{array}{ll} \max & \mathcal{L}(\mathbf{w}) = \sum_{m=1}^{M} \texttt{Acc}\left(\mathcal{P}(x_1^{(m)}, \cdots, x_6^{(m)}; \mathbf{w})\right) \\ \texttt{s.t.} & \texttt{approximated linear constraints} \end{array} \tag{4-8}$$

where $\mathbf{w} = \{w_{000}, w_{001}, \cdots, w_{666}\}$ and $w \in \mathbb{Z}$ are the CMBP parameters, M is the total number of turns of the training data, $\texttt{Acc}(\cdot)$ is the goal state tracking accuracy evaluation function. With the formulated optimisation problem (4-8), CMBP can be optimised as below:

1. Generate a superset of all feasible CMBP solutions satisfying the approximated linear constraints.

This generation can be regarded as a special kind of *integer linear programming* problem whose objective function is dummy. Existing integer linear programming solver can be used for this purpose. In this paper, SCIP ([23], Achterberg, 2009: 1–41.) is used. By setting additional constraints, the size of this superset can be controlled so that it is neither too small, nor too big.

2. $\mathcal{L}(\mathbf{w})$ is exhaustively calculated for each feasible solution from step 1.

   During the optimisation, due to relaxation of constraints, it is possible to get some $b_t(v)$ or b_t^r out of $[0, 1]$. To get legal track output, out-of-range $b_t(v)$ is always clipped to be 0 or 1 and the b_t^r is re-calculated accordingly.

3. Find the optimal CMBP solution by selecting the one with highest accuracy. Additional selection criterion, such as regularization terms, can also apply here.

With the above optimisation, the best integer coefficient CMBP can be found and this optimal solution can be refined when more training data is available.

Although CMBP was originally motivated from Bayesian probability operation which leads to the natural use of integer polynomial coefficient $w \in \mathbb{Z}$, CMBP can also be viewed as a statistical approach. Hence, the CMBP framework, equation (4-8), can be extended to *real* coefficient polynomials. The optimisation of real-coefficient CMBP is done by first getting an integer solution and then performing hill climbing search as shown in algorithm 1:

The above CMBP framework effectively bridges rule-based approaches and statistical approaches. The constraints reflect intuition prior knowledge and can be set manually, while the general Bayesian polynomial representation allows data-driven optimisation of model parameters. An additional advantage of the integer-programming based optimisation approach is that it is straightforward to find multiple feasible solutions satisfying constraints. It is then possible to perform system combination on the top N candidates of equation (4-8). In this paper, *belief score averaging* is investigated as a simple system combination approach.

Let $\mathbf{w} = \{w_{000}, \cdots, w_{666}\}$ be an integer coefficient solution set for equation (4-8);
Let $I \leftarrow \{i|w_i \in \mathbf{w}, w_i \neq 0\}$;
Let $D \leftarrow \{-0.4, -0.2, -0.1, 0.1, 0.2, 0.4\}$;
Let $\mathbf{w}^r \leftarrow \mathbf{w}$, *done* $\leftarrow$ **false**;
while *done is* **false do**
    done $\leftarrow$ **true**;
    foreach *index i in I* **do**
        foreach *step d in D* **do**
            Let $\Delta\mathbf{w} = \{\cdots, w_i + d, \cdots\}$;
            Let $\hat{\mathbf{w}}^r = \mathbf{w}^r + \Delta\mathbf{w}$;
            if $\mathcal{L}(\hat{\mathbf{w}}^r) > \mathcal{L}(\mathbf{w}^r)$ **then**
                $\mathbf{w}^r \leftarrow \hat{\mathbf{w}}^r$, *done* $\leftarrow$ **false**;
            end
        end
    end
end

Algorithm 1: Hill climbing algorithm for real coefficient solution

4.2 Experiment

As introduced in section 2, the DSTCs have provided the first common testbed in a standard format, along with a suite of evaluation metrics for dialogue state tracking ([7], Williams et al., 2013: 404–413.). In this paper, DSTC-2 and DSTC-3 tasks are used to evaluate the proposed approach. Both tasks provide training dialogues with turn-level ASR hypotheses, SLU hypotheses and user goal labels. DSTC-2 is a restaurant domain task and 2118 in-domain training dialogues are provided ([8], Henderson et al., 2014: 263–272.). While in DSTC-3, a tourist domain task, only 10 in-domain training dialogues are provided. The DSTC-3 task is to adapt the tracker trained on DSTC-2 data to the new domain with very few dialogues. In this section, only joint goal tracking is discussed. Table 4.1 is a summary of the size of datasets of DSTC-2 and DSTC-3.

The DST evaluation criteria are the *joint goal accuracy* and the *L2* ([8], Henderson et al., 2014: 263–272.)([9], Henderson et al., 2014: 324–329.). *Accuracy* is defined as the fraction of turns in which the tracker's 1-best joint goal hypothesis is correct, the larger the better. *L2* is the L2 norm between the distribution of all hypotheses output by the tracker and the correct goal distribution (a delta function), the smaller the better.

Table 4.1 Summary of data corpora of DSTC-2/3

Task	Dataset	#Dialogues	Usage
DSTC-2	`dstc2trn`	1612	Training
	`dstc2dev`	506	Training
	`dstc2eval`	1117	Test
DSTC-3	`dstc3seed`	10	Not used
	`dstc3eval`	2265	Test

Moreover, schedule 2 and labelling scheme A defined in Henderson et al. (2013) are used in both tasks. Specifically, schedule 2 only counts the turns where new information about some slots either in a system confirmation action or in the SLU list is observed. Labelling scheme A is that the labelled state is accumulated forwards through the whole dialogue. For example, the goal for slot s is "`None`" until it is informed as $s = v$ by the user, from then on, it is labelled as v until it is again informed otherwise.

Since the features of CMBP are all probabilistic features, the performance of CMBP is strongly correlated to the quality of confidence scores from SLU. It has been shown that the organiser-provided live SLU confidence was not good enough ([25], Zhu et al., 2014: 336–341.)([14], Sun et al., 2014: 318–326.). Hence, most of the state-of-the-art results from DSTC-2 and DSTC-3 used refined SLU (either explicitly rebuild a SLU component or take the ASR hypotheses into the trackers ([19], Williams, 2014: 282–291.)([14], Sun et al., 2014: 318–326.)([18], Henderson et al., 2014: 292–299.)([26], Henderson et al., 2014: 360–365.)([22], Kadlec et al., 2014: 348–353.)([10], Sun et al., 2014: 330–335.)). In accordance to this, except for the results directly taken from other papers (shown in table 4.5, 4.6, 5.4 and 5.5), all experiments in this paper used the output from a refined semantic parser ([25], Zhu et al., 2014: 336–341.)([14], Sun et al., 2014: 318–326.) instead of the live SLU provided by the organizer.

4.2.1 Investigation on CMBP Configurations

This section describes the experiments comparing different configurations of CMBP. All experiments were performed on the DSTC-2 tasks.

As indicated in section 4.1.2, multiple feasible solutions can be generated using integer programming, and the feasible solution space is controlled by the number of constraints so that it is neither too big nor too small. Table 4.2 compares the integer CMBP performance with different constraint sets[2] . For each constraint set, the detailed description can be found in the appendix. The number of feasible solutions is shown in column #Solutions, and the best integer CMBP is obtained by exhaustively checking the overall joint goal accuracy on the combined data set of `dstc2trn` and `dstc2dev`. The performance of the best CMBP is then evaluated on `dstc2eval`, shown in column Acc and L2.

Table 4.2 Performance of CMBP with different constraint sets on `dstc2eval`.

Constraint set	#Solutions	Acc	L2
{(6-23) - (6-34)}	7926	0.756	0.372
{(6-14),(6-23) - (6-34)}	461	0.756	0.370
{(6-14),(6-15),(6-23) - (6-34)}	132	0.756	0.375

It can be seen that larger solution space does not necessarily yield significantly better results. By applying more constraints (i.e. prior knowledge), the feasible CMBP space can be effectively controlled without losing much performance of the best CMBP contained in the space. In the following experiment, the constraint set is fixed to be $\{(6-14),(6-23)-(6-34)\}$.

Since multiple CMBP solutions can be used for combination, it is interesting to know whether the performance of the top N solutions are robust and comparable. Table 4.3 shows that the top 5 integer CMBP models have similar performance. This demonstrates the robustness of CMBP and implies that system combination is likely to be safe to yield improvement.

Table 4.3 Performance of top 5 CMBPs on `dstc2eval`.

Performance	N-Best CMBP Solution				
	1	2	3	4	5
Acc	0.756	0.756	0.756	0.756	0.756
L2	0.370	0.375	0.375	0.375	0.371

[2]The solution set is calculated by SCIP version 3.1.0 with 8 byte precision. Due to the limited numerical precision, the calculated solution may not be exactly the same as the real solution set.

The top-5 solutions in table 4.3 are found by purely optimising the overall goal accuracy of the training data (`dstc2trn+dstc2dev`). They usually have large complexity (i.e. the number of non-zero integer coefficients), for example, the 1-best solution has 12 parameters. As regularization, constraints in section 4.1.1 or terms in section 4.1.2, can be imposed on CMBP to control complexity and avoid over-fitting. To investigate the effect of regularisation, 12 CMBP models with different number of non-zero integer coefficients were randomly selected from the feasible solution space. The results are shown in Figure 4.1. Furthermore, the hill climbing algorithm is applied to each of them to generate corresponding optimal real-coefficient CMBPs.

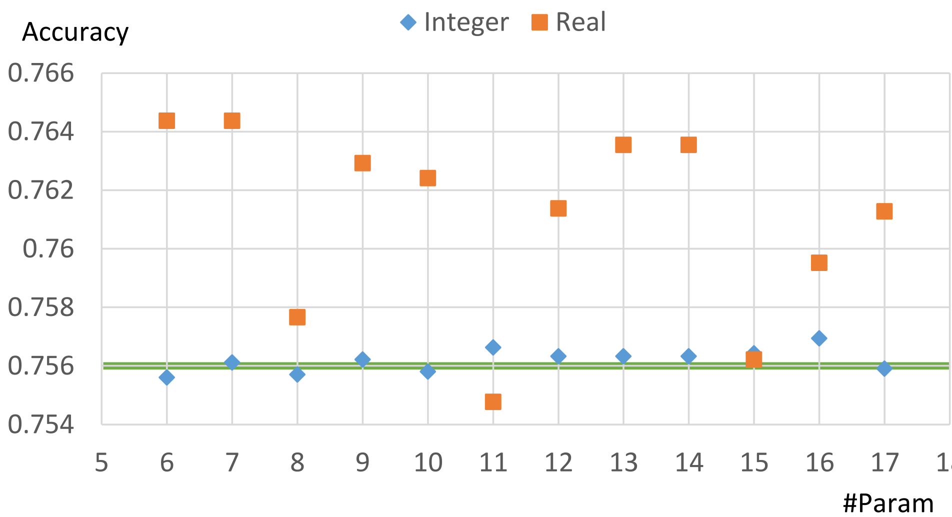

Figure 4.1 Performance of real-coefficient CMBPs compared with the corresponding integer-coefficient CMBPs on `dstc2eval` dataset

From Figure 4.1, the CMBPs with small complexity have similar performance compared to the top 5 integer CMBPs in table 4.3. This shows that by applying regularization term, the model size can be effectively reduced without hurting the performance. In addition, most of the real-coefficient CMBPs outperform the corresponding integer coefficient CMBPs, demonstrating the importance of extending integer coefficients to real numbers. Another observation is that there is no obvious correlation between the performance of the optimised real-coefficient CMBPs and the corresponding integer-coefficient CMBPs. In practice, the time algorithm 1 takes is positively correlated to the complexity of CMBP. Therefore, to efficiently optimise real-coefficient CMBPs, in the following experiments, we only run algorithm 1 on integer-coefficient CMBPs with the smallest number of non-zero coefficients.

It can be seen from Figure 4.1 that although in general the real-coefficient CMBPs have better performance, sometimes they do not outperform the corresponding integer-coefficient CMBPs. System combination is investigated here to make the performance of the real-coefficient CMBPs stable. The combined model, which applies belief score averaging on the the above 12 CMBPs, achieved an accuracy of 0.761 on `dstc2eval`. This is a very competitive result.

4.2.2 Comparison with Other DST Approaches

The previous section investigates how to get the best integer-coefficient CMBP and optimised real-coefficient CMBP, in this section, the performance of CMBP is compared to both rule-based and statistical approaches. As indicated before, CMBP can be naturally viewed as a data-driven approach with the probability features defined in 4.1.1, to make fair comparison, all statistical models in this section also use similar feature set. Altogether, 2 rule-based trackers and 2 statistical trackers were built for performance comparison:

- **MaxConf** is a *rule-based* model common used in spoken dialogue systems which always selects the value with the highest confidence score from the 1^{st} turn to the current turn. It was used as one of the primary baselines in DSTC-2 and DSTC-3.

- **HWU** is a *rule-based* model based on equation (3-3), proposed by Wang and Lemon (2013); Wang (2013). It is regarded as a simple, yet competitive baseline of DSTC-2 and DSTC-3.

- **DNN** is a *statistical model* with probability feature as CMBP. Since DNN does not have recurrent structures while RPN does, to fairly take into account this, the DNN feature set at the t^{th} turn is defined as

$$\bigcup_{i\in\{t-9,\cdots,t\}} \left\{P_i^+(v), P_i^-(v), \tilde{P}_i^+(v), \tilde{P}_i^-(v)\right\} \cup \left\{\hat{P}(t)\right\}$$

where $\hat{P}(t)$ is the highest confidence score from the 1^{st} turn to the t^{th} turn. The DNN has 3 hidden layers with 64 nodes per layer.

- **MaxEnt** is another *statistical model* using Maximum Entropy model with the same input feature as DNN.

In this section, the integer coefficient CMBPs for DSTC-2 were trained on `dstc2trn` and `dstc2dev`, and then algorithm 1 was used to extend it to real coefficient CMBPs. For DSTC-3, the CMBP model trained for DSTC-2 was directly used without modification. This means that the seed data of DSTC-3 was not used at all. This is to test the hypothesis that rule-based model is less sensitive to data compared to statistical approaches. Both 1-best integer-coefficient and optimised real-coefficient CMBPs are compared with the above DST approaches and the results are shown in table 4.4.

Table 4.4 Performance of 1-best integer-coefficient and optimised real-coefficient CMBPs compared with 2 baselines and 3 statistical trackers on the test dataset of DSTC-2 (`dstc2eval`) and DSTC-3 (`dstc3eval`).

Type	System	`dstc2eval`		`dstc3eval`	
		`Acc`	`L2`	`Acc`	`L2`
Rule	MaxConf	0.668	0.647	0.548	0.861
	HWU	0.720	0.445	0.594	0.570
Statistical	DNN	0.719	0.469	0.628	0.556
	MaxEnt	0.710	0.431	0.607	0.563
CMBP	Int	0.756	0.370	0.623	0.552
	Real	0.764	0.428	0.632	0.591

From the table it can been seen that with similar feature set, CMBP, especial real-coefficient CMBP, has relatively good performance compared with both rule-based approaches and statistical approaches.

4.2.3 Comparison with State-of-the-art DSTC Trackers

In DSTC-2 and DSTC-3, the state-of-the-art trackers mostly employed statistical approaches. Usually, richer feature set and more complicated model structures than the statistical models in section 4.2.2 are used. In this section, the proposed CMBP approach is compared to the best submitted trackers in DSTC-2/3, regardless of fairness of feature selection and the SLU refinement approach. The real-coefficient CMBP with score averaging of feasible CMBP solutions whose number of parameters is less than 8 was used here and the CMBP trained on DSTC-2 was again directly used for DSTC-3. The result is presented in table 4.5 and 4.6.

Table 4.5 Performance comparison between CMBP and best trackers of DSTC-2 on `dstc2eval`. Baseline* is the best results from the 4 baselines in DSTC2.

System	Approach	Rank	Acc	L2
Baseline*	Rule	5	0.719	0.464
Williams (2014)	LambdaMART	1	0.784	0.735
Henderson et al. (2014)	RNN	2	0.768	0.346
Sun et al. (2014)	DNN	3	0.750	0.416
CMBP	Sys. Comb.	2.5	0.762	0.436

Note that the Williams's system ([19], Williams, 2014: 282–291.) used batch ASR hypothesis information (i.e. off-line ASR re-decoded results) and can not be used in the normal on-line mode in practice. Hence, the practically best tracker is Henderson et al.([18], Henderson et al., 2014: 292–299.). It can be observed from table 4.5, CMBP ranks only second to the best practical tracker in accuracy, although the L2 performance is slightly worse. Since accuracy is the most important criterion, considering that CMBP used much simpler feature set and can operate very efficiently, it is quite competitive.

Table 4.6 Performance comparison between CMBP and best trackers of DSTC-3 on `dstc3eval`. Baseline* is the best results from the 4 baselines in DSTC3.

System	Approach	Rank	Acc	L2
Baseline*	Rule	6	0.575	0.691
Henderson et al. (2014)	RNN	1	0.646	0.538
Kadlec et al. (2014)	Rule	2	0.630	0.627
Sun et al. (2014)	Rule	3	0.610	0.556
CMBP	Sys. Comb.	1.5	0.634	0.579

It can be seen from table 4.6, CMBP trained on DSTC-2 can achieve state-of-the-art performance on DSTC-3. This demonstrates that CMBP successfully inherits the advantage of good generalisation ability of rule-based model.

Chapter 5 Recurrent Polynomial Network

5.1 Recurrent Polynomial Network

Recurrent polynomial network ([12], Sun et al., 2015: 1–22.) takes the other way to bridge rule-based and statistical approaches. The basic idea of RPN is to enable a kind of statistical model to take advantage of prior knowledge or intuition by using the parameters of rule-based models to initialize the parameters of statistical models.

Like common neural networks, RPN is a statistical approach so it is as easy to add features and try complex structures in RPN as in neural networks. However, compared with common neural networks which are "black boxes", an RPN can essentially be seen as a polynomial function. Hence, considering that a CMBP is also a polynomial function, the encoded prior knowledge and intuition in CMBP can be transferred to RPN by using the parameters of CMBP to initialize RPN skillfully. In this way, it bridges rule-based models and statistical models.

A recurrent polynomial network is a computational network. The network contains multiple edges and loops. Each node is either an *input node*, which is used to represent an input value, or a *computation node*. Each node x is set an initial value $u_x^{(0)}$ at time 0, and its value is updated at time $1, 2, \cdots$. Both the type of edges and the type of nodes decide how the nodes' values are updated. There are two types of edges. One type, referred to as *type-1*, indicates the value updating at time t takes the value of a node at time $t-1$, i.e. type-1 edges are recurrent edges, while the other type, referred to as *type-2*, indicates the value updating at time t takes another node's value at time t. For simplicity, let I_x be the set of nodes index y such which are linked to node x by a type-1 edge, $\hat{I}_x$ be the set of nodes y which are linked to node x by a type-2 edge. Based on these definitions, two types of computation nodes, *sum* and *product*, are introduced. Specifically, at time $t > 0$, if node x is a sum node, its value $u_x^{(t)}$ is updated by

$$u_x^{(t)} = \sum_{y \in I_x} w_{x,y} u_y^{(t-1)} + \sum_{y \in \hat{I}_x} \hat{w}_{x,y} u_y^{(t)} \tag{5-1}$$

where $w, \hat{w} \in \mathbb{R}$ are the weights of edges.

Similarly, if node x is a product node, its value is updated by

$$u_x^{(t)} = \prod_{y \in I_x} {u_y^{(t-1)}}^{M_{x,y}} \prod_{y \in \hat{I}_x} {u_y^{(t)}}^{\hat{M}_{x,y}} \tag{5-2}$$

where $M_{x,y}$ and $\hat{M}_{x,y}$ are integers, denoting the multiplicity of the type-1 edge $\overrightarrow{yx}$, and the multiplicity of the type-2 edge $\overrightarrow{yx}$ respectively. It is noted that only $w, \hat{w}$ are parameters of RPN while $M_{x,y}$, $\hat{M}_{x,y}$ are constant given the structure of an RPN.

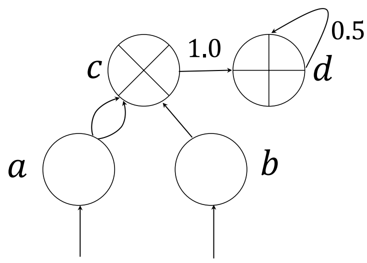

Figure 5.1 A simple example of RPN. The type of node a, b, c, d is input, input, product, and sum respectively. Edge $\overrightarrow{dd}$ is of type-1, while the other edges are of type-2. $\hat{M}_{a,c} = 2$, $\hat{M}_{b,c} = \hat{M}_{c,d} = M_{d,d} = 1$.

Let $\boldsymbol{u}^{(t)}$, $\hat{\boldsymbol{u}}^{(t)}$ denote the vector of computation nodes' values and the vector of input nodes' values at time t respectively, then a well-defined RPN can be seen as a polynomial function as below.

$$\boldsymbol{u}^{(t)} = \mathcal{P}\left(\hat{\boldsymbol{u}}^{(t)} \oplus \boldsymbol{u}^{(t-1)} \oplus \hat{\boldsymbol{u}}^{(t-1)} \oplus \{1\}\right) \tag{5-3}$$

where $\oplus$ denotes vector concatenation and $\mathcal{P}$ is defined by equation (4-2). For example, for the RPN in figure 5.1, its corresponding polynomial function is

$$\begin{aligned}(u_c^{(t)}, u_d^{(t)}) =& \mathcal{P}\left(u_a^{(t)}, u_b^{(t)}, u_c^{(t-1)}, u_d^{(t-1)}, u_a^{(t-1)}, u_b^{(t-1)}\right) \\ =& \left({(u_a^{(t)})}^2 u_b^{(t)}, 0.5u_d^{(t-1)} + {(u_a^{(t)})}^2 u_b^{(t)}\right)\end{aligned} \tag{5-4}$$

Each computation node can be regarded as an *output node*. For example, for the RPN in figure 5.1, node c and node d can be set as output nodes.

5.2 RPN for Dialogue State Tracking

As introduced in section 2, in this paper, the dialogue state tracker receives SLU N-best hypotheses for each user turn, each hypothesis having a set of act-slot-value tuples with a confidence score. The dialogue state tracker is supposed to output a set of distributions of the joint user goal, that is, the value for each slot. For simplicity and consistency with chapter 4 and the work of Sun et al. (2014) and Yu et al. (2015), slot and value independence are assumed in the RPN model for dialogue state tracking, though neither CMBP nor RPN is limited to the assumptions. Besides, in the rest of the paper, $b_t(v), P_t^+(v), P_t^-(v), \tilde{P}_t^+(v), \tilde{P}_t^-(v)$ are abbreviated by $b_t, P_t^+, P_t^-, \tilde{P}_t^+, \tilde{P}_t^-$ respectively in circumstances where there is no ambiguity.

5.2.1 Basic Structure

Before describing details of the structure used in the real situations, to help understand the corresponding relationship between RPN and CMBP, let's first look at a simplified case with a smaller feature set and a smaller order, which is a corresponding relationship between the RPN shown in figure 5.2 and 2-order polynomial (5-5) with features $b_{t-1}, P_t^+, 1$:

$$\begin{aligned} b_t &= 1 - (1 - b_{t-1})(1 - P_t^+) \\ &= b_{t-1} + P_t^+ - P_t^+ b_{t-1} \end{aligned} \tag{5-5}$$

Recall that a CMBP of polynomial order 2 with 3 features is the following equation (refer to equation (4-2)):

$$\mathcal{P}(\iota_0, \iota_1, \iota_2) = \sum_{0 \le k_1 \le k_2 \le 2} g_{k_1,k_2} \prod_{1 \le i \le 2} \iota_{k_i} \tag{5-6}$$

The RPN in figure 5.2 has three layers. The first layer only contains input nodes. The second layer only contains product nodes. The third layer only contains sum nodes. Every product node in the second layer denotes a monomial of order 2 such as $(b_{t-1})^2, b_{t-1}, P_t^+$ and so on. Every product node in the second layer is linked to the sum node in the third layer whose value is a weighted sum of value of product nodes.

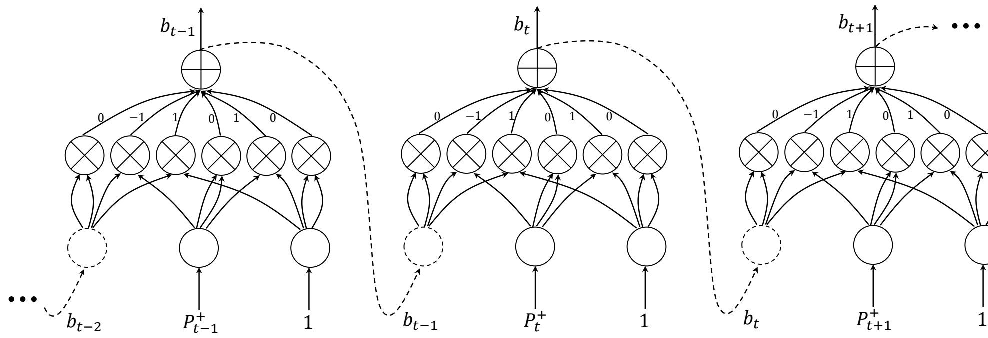

Figure 5.2 A simple example of RPN for DST.

With weight set according to coefficients in equation (5-5), the value of sum node in the third layer is essentially the b_t in equation (5-5).

Like the above simplified case, a layered RPN structure shown in figure 5.3 is used for dialogue state tracking in our first trial which essentially corresponds to 3-order CMBP, though the RPN framework is not limited to the layered topology. Recall that a CMBP of polynomial order 3 is used as shown in the following equation (refer to equation (4-2)):

$$\mathcal{P}(\iota_0, \cdots, \iota_D) = \sum_{0 \leq k_1 \leq k_2 \leq k_3 \leq D} g_{k_1,k_2,k_3} \prod_{1 \leq i \leq 3} \iota_{k_i} \tag{5-7}$$

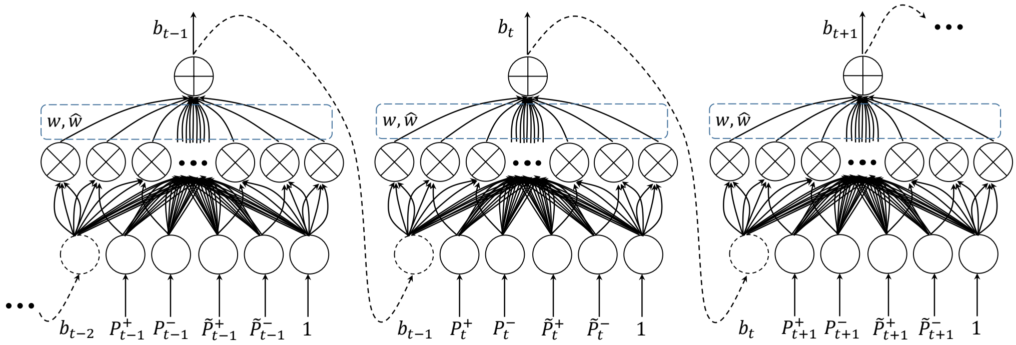

Figure 5.3 RPN for DST.

Let (l, i) denote the index of i-th node in the l-th layer. The detailed definitions of each layer are as follows:

- First layer / Input layer:

Input nodes are features at turn t, which corresponds to variables in CMBP in section 4.1.1. i.e.

- $u^{(t)}_{(1,0)} = b_{t-1}$
- $u^{(t)}_{(1,1)} = P^+_t$
- $u^{(t)}_{(1,2)} = P^-_t$
- $u^{(t)}_{(1,3)} = \tilde{P}^+_t$
- $u^{(t)}_{(1,4)} = \tilde{P}^-_t$
- $u^{(t)}_{(1,5)} = 1$

While 7 features are used in chapter 4 and previous work of CMBP ([10], Sun et al., 2014: 330–335.)([11], Yu et al., 2015: 1–10.), only 6 of them are used in RPN with feature b^r_{t-1} removed (b^r_t is defined in section 4.1.1). Since our experiments showed the performance of CMBP would not become worse without feature b^r_{t-1}, to make the structure more compact, b^r_{t-1} is not used in this paper for RPN. In accordance to this, CMBP mentioned in the rest of paper does not use this feature either.

- Second layer:

The value of every product node in the second layer is a monomial like the simplified case. And every product node has indegree 3 which is corresponding to the order of CMBP.

Every monomial in CMBP is the product of three repeatable features. Correspondingly, the value of every product node in second layer is the product of values of three repeatable nodes in the first layer. Every triple $(k_1, k_2, k_3)(0 \le k_1 \le k_2 \le k_3 \le 5)$ is enumerated to create a product node $x = (2, i)$ in second layer that nodes $(1, k_1), (1, k_2), (1, k_3)$ are linked to. i.e. $\hat{I}_x = \{(1, k_1), (1, k_2), (1, k_3)\}$. And thus $u^{(t)}_x = u^{(t)}_{1,k_1} u^{(t)}_{1,k_2} u^{(t)}_{1,k_3}$.

And different node in the second layer is created by a distinct triple. So given the 6 input features, there are $\sum\limits_{k_1=0}^{5} \sum\limits_{k_2=k_1}^{5} \sum\limits_{k_3=k_2}^{5} 1 = \binom{6+3-1}{3} = 56$ nodes in the second layer.

To simplify the notation, a bijection from nodes to monomials is defined as:

$$\mathcal{F}: \{x|x \text{ is the index of a node in the } 2^{nd} \text{ layer}\} \rightarrow \{(k_1, k_2, k_3)|0 \le k_1 \le k_2 \le k_3 \le D\} \tag{5-8}$$

$$\mathcal{F}(x) = (k_1, k_2, k_3) \Longleftrightarrow u_{2,i}^{(t)} = u_{1,k_1}^{(t)} u_{1,k_2}^{(t)} u_{1,k_3}^{(t)} \tag{5-9}$$

where $D + 1 = 6$ is the number of nodes in the first layer, i.e. input feature dimension.

- Third layer:

  The value of sum node $x = (3, 0)$ in the third layer is corresponding to the output value of CMBP.

  Every product nodes in the second layer are linked to it. Node x's value $u_{3,0}^{(t)}$ is a weighted sum of values of product node $u_{2,i}^{(t)}$ where the weights correspond to g_{k_1,k_2,k_3} in equation (5-7).

With only sum and product operation involved, every node's value is essentially a polynomial of input features. And just like recurrent neural network, node at time t can be linked to node at time $t + 1$. That is why this model is called recurrent polynomial network.

The parameters of the RPN can be set skillfully according to CMBP coefficients g_{k_1,k_2,k_3} in equation (5-7) so that the output value is the same as the value of CMBP, which is a direct way of applying prior knowledge and intuition to statistical models. It is explained in detail in section 5.2.4.

5.2.2 Activation Function

In DST, the output value is a belief which should lie in $[0, 1]$, while values of computational nodes are not bound by certain interval in RPN. Experiments showed that if weights are not properly set in RPN and a belief b_{t-1} output by RPN is larger than 1, then b_t may grow much larger because b_t is the weighted sum of monomials such as

$(b_{t-1})^3$. Belief of later turns such as b_{t+10} may grow to a number which is so large that can hardly be calculated.

Therefore, an activation function is needed to map b_t to a legal belief value (referred to as b'_t) in $(0, 1)$. 3 kinds of functions, the *logistic* function, the *clip* function, and the *softclip* function have been considered. A logistic function is defined as

$$logistic(x) = \frac{L}{1 + e^{-\eta(x - x_0)}} \tag{5-10}$$

It can map $\mathbb{R}$ to $(0, 1)$ by setting $L = 1$. However, even with carefully setting η and x_0, such as $\eta = 5, x_0 = 0.5$, the gap between b'_t and b_t can hardly be omitted when b_t is in the range of $(0, 1)$ so that it makes RPN inherit the prior knowledge from CMBP more difficult. For example, if logistic function is used and P_t^+, P_t^-, $\tilde{P}_t^+$, $\tilde{P}_t^-$ are all 0 at some turn t. If b_{t-1} is in $[0, 1]$ and activation function is linear on $[0, 1]$, using the constraints given by the previous work of CMBP ([10], Sun et al., 2014: 330–335.)([11], Yu et al., 2015: 1–10.), with certain parameter set to constant, it is easily ensured that $b_t = b_{t-1}$. However, constraints in CMBP should be changed to achieve this property if logistic function is used.

As an alternation, a *clip* function is defined as

$$clip(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } 0 \leq x \leq 1 \\ 1 & \text{if } x > 1 \end{cases} \tag{5-11}$$

It is linear on $[0, 1]$. However, if $b'_t = clip(b_t)$, $b_t \notin [0, 1]$ and $\mathcal{L}$ is the loss function,

$$\frac{\partial \mathcal{L}}{\partial b_t} = \frac{\partial \mathcal{L}}{\partial b'_t}\frac{\partial b'_t}{\partial b_t} = \frac{\partial \mathcal{L}}{\partial b'_t} \times 0 = 0 \tag{5-12}$$

Thus, $\frac{\partial \mathcal{L}}{\partial b_t}$ would be 0 whatever $\frac{\partial \mathcal{L}}{\partial b'_t}$ is. This gradient vanishing phenomenon may affect the effectiveness of backpropagation training in section 5.2.5.

So an activation function $softclip(\cdot)$ is introduced, which is a combination of logistic function and clip function. Let ϵ denote a small value such as 0.01, δ denote

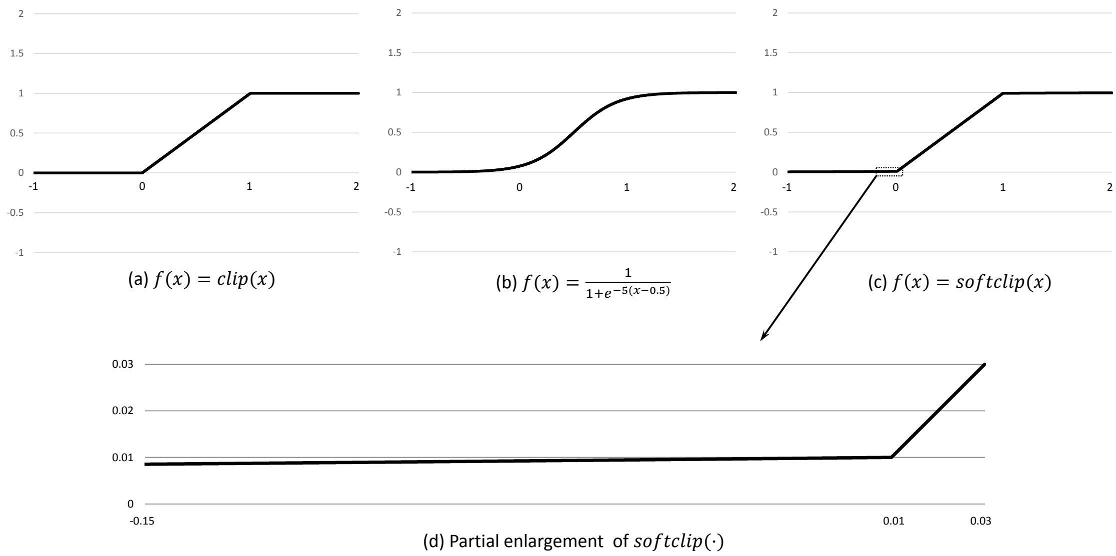

Figure 5.4 Comparison among clip function, logistic function, and softclip function

the offset of sigmoid function such that $sigmoid\,(\epsilon - 0.5 + \delta) = \epsilon$. Here the *sigmoid* function refers to the special case of the logistic function defined by the formula

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{5-13}$$

The softclip function is defined as

$$softclip(x) \triangleq \begin{cases} sigmoid\,(x - 0.5 + \delta) & \text{if } x \leq \epsilon \\ x & \text{if } \epsilon < x < 1 - \epsilon \\ sigmoid\,(x - 0.5 - \delta) & \text{if } x \geq 1 - \epsilon \end{cases} \tag{5-14}$$

$softclip : \mathbb{R} \rightarrow (0, 1)$ is a non-decreasing, continuous function. However, It is not differentiable when $x = \epsilon$ or $x = 1 - \epsilon$. So we defined its derivative as follows:

$$\frac{\partial softclip(x)}{\partial x} \triangleq \begin{cases} \frac{\partial sigmoid(x - 0.5 + \delta)}{\partial x} & \text{if } x \leq \epsilon \\ 1 & \text{if } \epsilon < x < 1 - \epsilon \\ \frac{\partial sigmoid(x - 0.5 - \delta)}{\partial x} & \text{if } x \geq 1 - \epsilon \end{cases} \tag{5-15}$$

It is like a clip function. However, its derivative may be small on some inputs but is not zero. Figure 5.4 shows the comparison among clip function, logistic function, and softclip function.

With the activation function, a new type of computation node, referred to as *activation node*, is introduced. Activation node only takes one input and only has one input edge of type-2, i.e $|\hat{I}_x| = 1$ and $I_x = \emptyset$. The value of an activation node x is calculated as

$$u_x^{(t)} = softclip\left(u_{\jmath_x}^{(t)}\right) \tag{5-16}$$

where $\jmath_x$ denotes the input node of node x. i.e. $\hat{I}_x = \{\jmath_x\}$.

The activation function is used in the rest of the paper. Figure 5.5 gives an example of RPN with activation function, whose structure is constructed by adding an activation function to the RPN in figure 5.3.

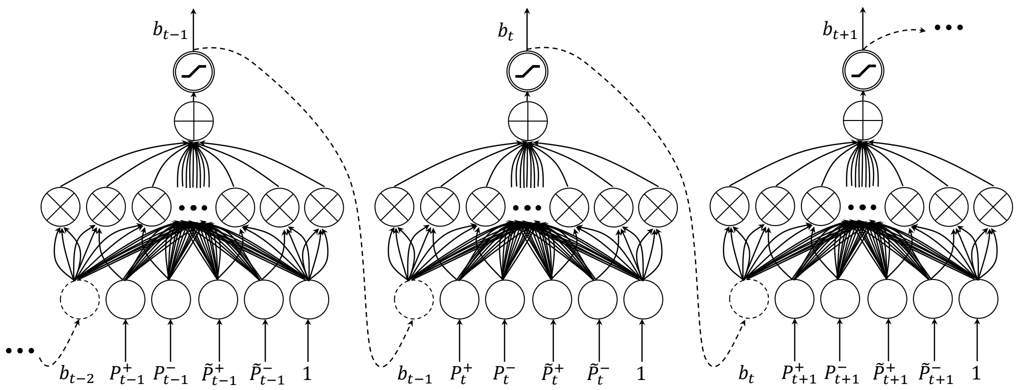

Figure 5.5 RPN for DST with activation functions

5.2.3 Complete Structure

Adding features to CMBP is not easy because additional prior knowledge is needed to add to keep the search space not too large. Concretely, adding features can introduce new monomials. Since the trivial search space is exponentially increasing as the number of monomials, the search space tends to be too large to explore when new features are added. Hence, to reduce the search space, additional prior knowledge is needed, which can introduce new constraints to the polynomial coefficients. For the same reason, increasing the model complexity is also not very convenient in CMBP.

In contrast to that, since RPN can be seen as a statistically model, it is as easy as most statistical approaches such as RNN to add new features to RPN and use more complex structures. At the same time, no matter what new features are used and how complex the structure is, RPN can always take advantage prior knowledge and intuition which is discussed in section 5.2.4. In this paper, both new features and complex structure are explored.

Adding new features can be done by just adding input nodes which correspond to the new features, and then adding product nodes corresponding to the new possible monomials introduced by the new input nodes. In this paper, for slot s, value v at turn t, in addition to $f_0 \sim f_5$ which are defined as $b_{t-1}(v)$, $P_t^+(v)$, $P_t^-(v)$, $\tilde{P}_t^+(v)$, $\tilde{P}_t^-(v)$, and 1 respectively, 4 new features are investigated. f_6 and f_7 are features of system acts at the last turn:

- $f_6 \triangleq canthelp(s,t,v) \cup canthelp.missing\_slot\_value(s,t)$ =1 if the system cannot offer a venue with the constraint $s = v$ or the value of slot s is not known for the selected venue, otherwise 0.

- $f_7 \triangleq select(s,t,v)$ =1 if the system asks the user to pick a suggested value for slot s, otherwise 0.

f_6 and f_7 are introduced because user is likely to change their goal if given machine acts $canthelp(s,v)$, $canthelp.missing\_slot\_value(s,t)$ and $select(s,v)$. f_8 and f_9 are features of user acts at the current turn:

- $f_8 \triangleq inform(s,t,v) = 1$ if one of SLU hypotheses to the user is informing slot s is v, otherwise 0.

- $f_9 \triangleq deny(s,t,v)$ =1 if one of SLU hypotheses to the user is denying slot s is v, otherwise 0.

f_8 and f_9 are features about SLU acttype, introduced to make system robust when the confidence scores of SLU hypothesis are not reliable.

In this paper, the complexity of evaluating and training RPN for DST would not increase sharply because a constant order 3 is used and number of product nodes in the second layer grows from 56 to 220 when number of features grows from 6 to 10.

In addition to new features, RPN of more complex structure is also investigated in this paper. To capture some property just like belief b_t of dialogue process, a new sum node $x = (3, 1)$ in the third layer is introduced. The connection of $(3, 1)$ is the same as $(3, 0)$, so it introduces a new recurrent connection. The exact meaning of its value is unknown. However, it is the only value used to record information other than b_t of previous turns. Every other input features except b_t are features of current turn t. Compared with b_t, there are fewer restrictions on the value of $(3, 1)$ since its value is not directly supervised by the label. Hence, introducing $(3, 1)$ may help to reduce the effect of inaccurate labels.

The structure of the RPN with 4 new features and 1 new sum node, together with new activation nodes introduced in section 5.2.2 is shown in figure 5.6.

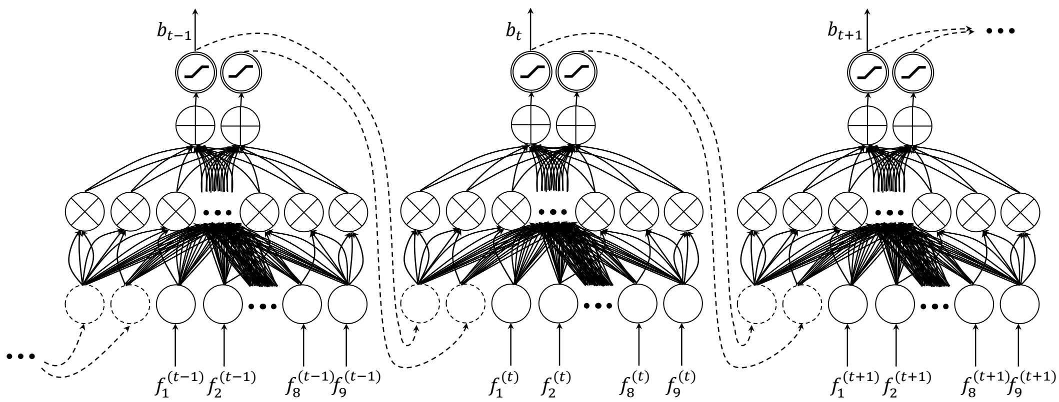

Figure 5.6 RPN with new features and more complex structure for DST

5.2.4 RPN Initialization

Like most neural network models such as RNN, the initialization of RPN can be done by setting each weight, i.e. w and $\hat{w}$, to be a small random value. However, with its unique structure, the initialization can be much better by taking advantage of the relationship between CMBP and RPN which is introduced in section 5.2.1.

When RPN is initialized according to a CMBP, prior knowledge and constraints are used to set RPN's initial parameters as a suboptimum point in the whole parameter space. RPN as a statistical model can fully utilize the advantages of statistical approaches. However, RPN is better than real CMBP while they both use data samples to train parameters. In the work of Yu et al. (2015), real-coefficient CMBP uses hill climbing to adjust parameters that are initially not zero and the change of parameters are always a multiple of 0.1. RPN can adjust all parameters including parameters initialized as 0 concurrently, while the complexity of adjusting all parameters concurrently is nearly the same as adjusting one parameter in CMBP. Besides, the change of parameters can be large or small, depending on learning rate. Thus, RPN and CMBP both are bridging rule-based models and statistical ones, while RPN is a statistical model utilizing rule advantages and CMBP is a rule model utilizing statistical advantages.

In fact, given a CMBP, an RPN can achieve the same performance as the CMBP just by setting its weights skillfully according to the coefficients of the CMBP. To illustrate that, the steps of initializing the RPN in figure 5.6 with a CMBP of features $f_0 \sim f_9$ is described below.

First, to ensure that the new added sum node $x = (3, 1)$ will not influence the output b_t in RPN with initial parameters, $\hat{w}_{x,y}$ is set to 0 for all y. So node x's value $u_x^{(t)}$ is always 0.

Next, considering the RPN in figure 5.6 has more features than CMBP does, the weights related the new features should be set to 0. Specifically, suppose node x is the sum node in the third layer in RPN denoting b_t before activation and node y is one of the product nodes in the second layer denoting a monomial, if product node y is products of features f_6, f_7, f_8, f_9 or the added sum node, then node y's value is not a monomial in CMBP, then weights $\hat{w}_{x,y}$ should be set to 0.

Finally, if product node y is the product of features $f_0 \sim f_5$, suppose the order of CMBP is 3, then $\mathcal{F}(y) = (k_1, k_2, k_3)$ defined in equation (5-8) should satisfy $0 \leq k_1 \leq k_2 \leq k_3 \leq 5$. Weights $\hat{w}_{xy}$ should be initialized as g_{k_1,k_2,k_3} which is the coefficient of

$f_{k_1} f_{k_2} f_{k_3}$ in CMBP. Thus,

$$w_{x,y} = \begin{cases} g_{k_1,k_2,k_3} & \text{if } x = (2,0) \text{ and } \mathcal{F}(x) = (k_1, k_2, k_3) \\ 0 & \text{otherwise} \end{cases} \tag{5-17}$$

For RPN of other structures, the initialization can be done by following similar steps.

Experiments show that after training, there are only a few weights larger than 0.1, no matter using CMBP or random initialization.

5.2.5 Training RPN

For a slot s, value v, and time t, suppose l_t is the indicator of goal $s = v$ being part of joint goal at turn t in the dialogue label.

Suppose node x is the output node at turn t, and $u_x^{(t)}$ is the output value at turn t. If the mean squared error (MSE) is used as the training criterion and there are T turns, the cost $\mathcal{L}$ is

$$\mathcal{L} = \frac{1}{T} \sum_{i=1}^{T} (u_x^{(t)} - l_t)^2 \tag{5-18}$$

Forward Pass For each training sample, every node's value at every time is evaluated first. When evaluating $u_x^{(t)}$, values of nodes in I_x and $\hat{I}_x$ should be evaluated before. The computation formula should be based on the type of node x. In particular, for a layered RPN structure, we can simply evaluate $u_{x_1}^{t_1}$ earlier than $u_{x_2}^{t_2}$ if $t_1 < t_2$ or $t_1 = t_2$ and x_1's layer number is smaller than x_2's.

Backward Pass Backpropagation through time (BPTT) is used in training RPN. Let error of node x at time t $\delta_x^{(t)} = \frac{\partial \mathcal{L}}{\partial u_x^{(t)}}$. If a node x is an output node, then $\delta_x^{(t)}$ should be set according to its label l_t and output value $u_x^{(t)}$, otherwise $\delta_x^{(t)}$ should be initialized to 0. After a node's error $\delta_x^{(t)}$ is determined, it can be passed to $\delta_y^{(t-1)} (y \in I_x)$ and $\delta_y^{(t)} (y \in \hat{I}_x)$. Error passing should follow the reversed edge's direction. So the order of

Initialize $\Delta w_{xy} = 0, \Delta \hat{w}_{x,y} = 0$ for every x, y
Initialize the value of recurrent node at turn 0 as 0
foreach *Training sample slot s, value v* **do**
    for $t \leftarrow 1$ **to** T **do**
        for $d \leftarrow 1$ **to** 4 **do**
            foreach *node x in time t, layer d* **do**
                evaluate $u_x^{(t)}$
                if *x is output node* **then**
                    $\delta_x^{(t)} \leftarrow 2(u_x^{(t)} - l_t)$
                else
                    $\delta_x^{(t)} \leftarrow 0$
    for $t \leftarrow T$ **to** 1 **do**
        for $d \leftarrow 4$ **to** 1 **do**
            foreach *node x in time t, layer d* **do**
                foreach *node $y \in \hat{I}_x$* **do**
                    $\delta_y^{(t)} \leftarrow \delta_y^{(t)} + \delta_x^{(t)} \frac{\partial u_x^{(t)}}{\partial u_y^{(t)}}$
                foreach *node $y \in I_x$* **do**
                    $\delta_y^{(t-1)} \leftarrow \delta_y^{(t-1)} + \delta_x^{(t)} \frac{\partial u_x^{(t)}}{\partial u_y^{(t-1)}}$
    for $t \leftarrow 1$ **to** T **do**
        for $d \leftarrow 1$ **to** 4 **do**
            foreach *node x in time t, layer d* **do**
                foreach *node $y \in \hat{I}_x$* **do**
                    $\Delta \hat{w}_{xy} \leftarrow \Delta \hat{w}_{xy} + \alpha \delta_x^{(t)} u_y^{(t)}$
                foreach *node $y \in I_x$* **do**
                    $\Delta w_{xy} \leftarrow \Delta w_{xy} + \alpha \delta_x^{(t)} u_y^{(t-1)}$
foreach *edge(x, y)* **do**
    $w_{xy} \leftarrow w_{xy} - \Delta w_{xy}$
    $\hat{w}_{xy} \leftarrow \hat{w}_{xy} - \Delta \hat{w}_{xy}$

Algorithm 2: Training Algorithm of RPN for DST nodes passing error can follow the reverse order of evaluating nodes' values.

When every $\delta_x^{(t)}$ has been evaluated, the increment on weight $\hat{w}_{xy}$ can be calculated by

$$\begin{aligned}\Delta\hat{w}_{xy} &= \alpha\frac{\partial\mathcal{L}}{\partial\hat{w}_{xy}} \\ &= \alpha\sum_{i=1}^{T}\frac{\partial\mathcal{L}}{\partial u_x^{(t)}}\frac{\partial u_x^{(t)}}{\partial\hat{w}_{xy}} \\ &= \alpha\sum_{i=1}^{T}\delta_x^{(t)}u_y^{(t)}\end{aligned} \tag{5-19}$$

where α is the learning rate. Δw_{xy} can be evaluated similarly.

Note that only w_{xy} and $\hat{w}_{xy}$ are parameters.

The complete formula of evaluating node value $u_x^{(t)}$ and passing error $\delta_x^{(t)}$ can be found in appendix.

In this paper, full batch is used in training RPN for DST. In each training epoch, Δw_{xy} and $\Delta\hat{w}_{xy}$ are calculated for every training sample and added together. The weight w_{xy} and $\hat{w}_{xy}$ is updated by

$$w_{xy} = w_{xy} - \Delta w_{xy} \tag{5-20}$$

$$\hat{w}_{xy} = \hat{w}_{xy} - \Delta\hat{w}_{xy} \tag{5-21}$$

The pseudocode of training is shown in algorithm 2.

5.2.6 Complex Structure

In this paper, to search RPN's power of utilizing more features, multiple activation functions and a deeper structure, two interesting explorations on RPN structure are shown in this section, although they do not yield better results.

Complex Structure

Firstly, to express a 4-order polynomial, simply using the structure shown in figure 5.6 with in-degree of nodes in second layer increased to 4 would be sufficient. However, it can be expressed by a more compact RPN structure. To simplify the explanation, the example RPN expressing $1-(1-(b_{t-1})^2)(1-(P_t^+)^2)$ is shown in figure 5.7.

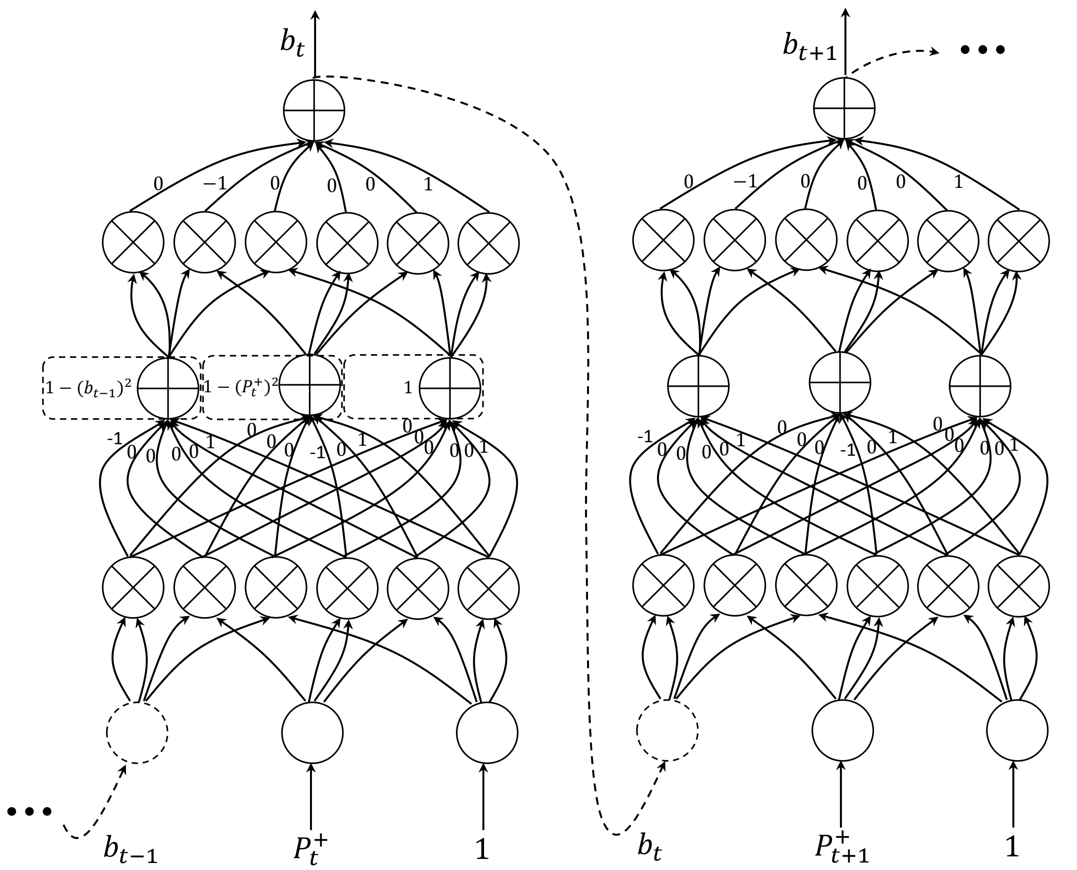

Figure 5.7 RPN for polynomial $1-(1-(b_{t-1})^2)(1-(P_t^+)^2)$

In figure 5.7, the first layer is used for input, values of product nodes in the second layer equal to products of two features such as $(b_{t-1})^2$, $b_{t-1}P_t^+$, $(P_t^+)^2$ and so on. Every sum node in the third layer can express all the possible 2-order polynomial of features with weights set accordingly. In figure 5.7, the values of the three sum nodes are $1-(b_{t-1})^2$, $1-(P_t^+)^2$ and 1 respectively. Then similarly, with another product nodes layer and sum nodes layer, the value of the output node in the last layer equals the value of the 4-order polynomial $(1-(b_{t-1})^2)(1-(P_t^+)^2)$.

The complete RPN structure with same features shown in figure 5.6, the new recurrent connection and activation nodes that expresses 4-order CMBPs can be obtained similarly.

With limited sum nodes in the third layer, the complexity of the model is much smaller than using a structure shown in figure 5.6 with product node's in-degree increased to 4 and increasing the number of product nodes accordingly.

Complex Features

Secondly, RNN proposed by Henderson et al. (2014) uses n-gram of ASR results and machine acts. Similar to that, features of n-gram of ASR results and machine acts are also investigated in RPN. Since RPN used in this paper is a binary classification model and assumes slots independent of each other, the n-gram features proposed by Henderson et al. (2014) are slightly modified in this paper by removing/merging some features to make the features independent of slots and values. For example, given machine acts `hello() | inform(area=center) | inform(food=Chinese) | request(name)`, for slot *food* and value *Chinese*, the n-gram machine act features are `hello`, `inform`, `request`, `inform+slot`, `inform+value`, `inform+slot+value`, `slot`, `value`, `slot+value`.

To combine RPN with RNN proposed by Henderson et al. (2014), input nodes of these n-gram features are not linked to product nodes in the second layer. Instead, a layer of sum nodes followed by a layer of activation nodes with sigmoid activation function, which are equivalent to a layer of neurons are introduced. And these activation nodes are linked to sum nodes in the third layer just like product nodes in the second layer. The structure is illustrated by figure 5.8 clearly.

Experiments have shown that these two structures do not yield better results when initialized randomly or initialized using 3-order CMBPs, although the model complexity increases a lot ([28], Xie et al., 2015: 1–9.). This indicates the briefness and effectiveness of the simple structure shown in figure 5.6.

5.3 Experiment

As mentioned in section 4.2, it has been shown that the organiser-provided live SLU confidence was not good enough, the output from a refined semantic parser is used instead of the live SLU provided by the organizer. Besides, the datasets and evaluation

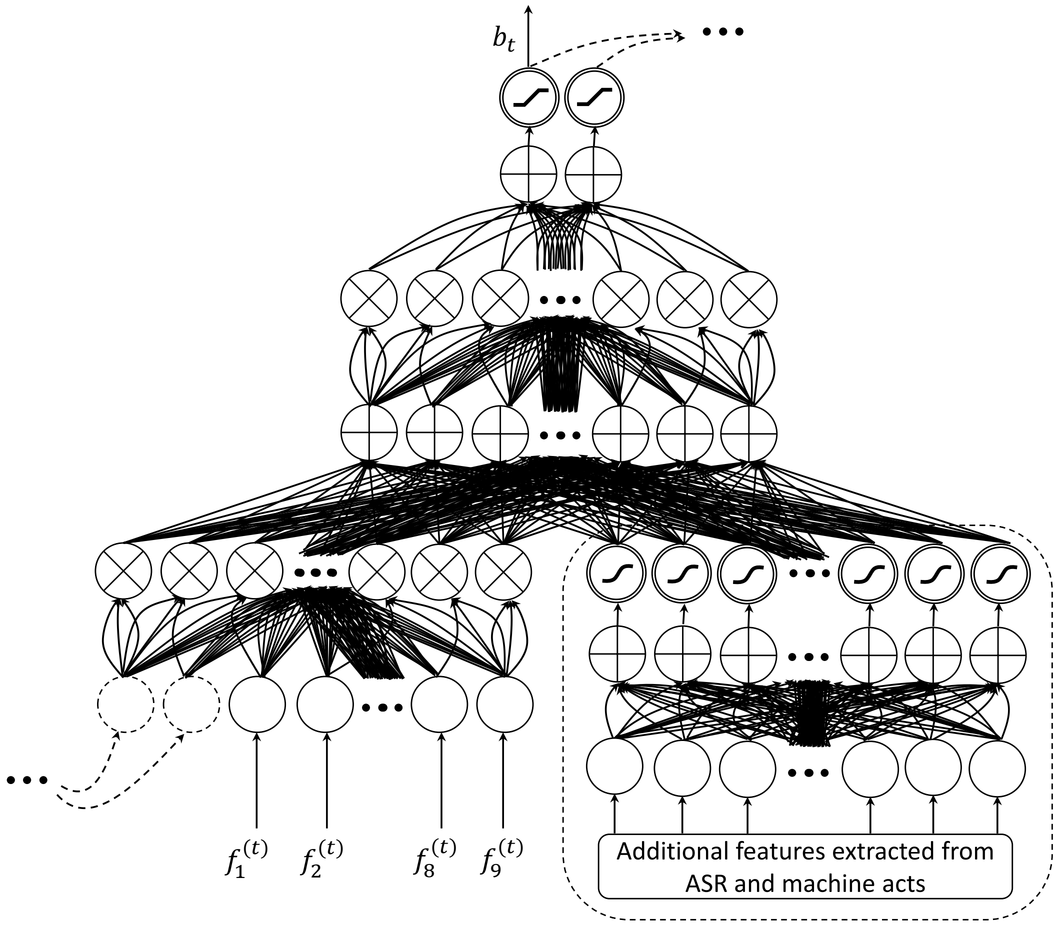

Figure 5.8 RPN structure combined with RNN features and structures

criteria are the same as section 4.2.

For all experiments, MSE is used as the training criterion and full-batch batch is used. For both DSTC-2 and DSTC-3 tasks, `dstc2trn` and `dstc2dev` are used, 60% of the data is used for training and 40% for validation. Validation is performed every 5 epochs. Learning rate is set to 1.0 initially. During the training, learning rate is halved when the MSE starts increasing. Training is stopped when the learning rate is sufficiently small, or the maximum number of training epochs is reached. Here, the maximum number of training epochs is set to 250.

5.3.1 Investigation on RPN Configurations

This section describes the experiments comparing different configurations of RPN. All experiments were performed on both the DSTC-2 and DSTC-3 tasks.

As indicated in section 5.2.4, an RPN can be initialized by a CMBP. Table 5.1 shows the performance comparison between initialization with a CMBP and with random values. In this experiment, the structure shown in figure 5.5 is used.

Table 5.1 Performance comparison between the RPN initialized by random values, and the RPN initialized by the CMBP coefficients on `dstc2eval` and `dstc3eval`.

Initialization	dstc2eval		dstc3eval	
	Acc	L2	Acc	L2
Random	0.722	0.435	0.500	0.671
CMBP	0.758	0.370	0.644	0.542

The performance of the RPN initialized by random values sampled from $\mathcal{N}(0, 0.01)$ is compared with the performance of the RPN initialized by the integer-coefficient CMBP. Here, the CMBP has 11 non-zero coefficients and has the best performance in DSTC-2. It can be seen from table 5.1 that the RPN initialized by the CMBP coefficients significantly outperforms the RPN initialized by random values. This demonstrates the encoded prior knowledge and intuition in CMBP can be transferred to RPN to improve RPN's performance, which is one of RPN's advantage, bridging rule-based models and statistical models. In the rest of the experiments, all RPNs use CMBP coefficients for initialization.

Since section 5.2.3 shows that it is convenient to add features and try more complex structures, it is interesting to investigate RPNs with different feature sets and structures, as shown in table 5.2. It can be seen that while no obvious correlation between the performance and different configurations of feature sets and structures can be observed on `dstc2eval`, new features and new recurrent connections significantly help improve the performance of RPN on `dstc3eval`. Thus, in the rest of the paper, both new features and new recurrent connections are used in RPN, unless otherwise stated.

Table 5.2 Performance comparison among RPNs with different configurations on `dstc2eval` and `dstc3eval`.

Feature Set	New Recurrent Connections	`dstc2eval`		`dstc3eval`	
		`Acc`	`L2`	`Acc`	`L2`
$f_0 \sim f_5$	No	0.758	0.370	0.644	0.542
$f_0 \sim f_9$		0.755	0.374	0.646	0.541
$f_0 \sim f_5$	Yes	0.757	0.372	0.645	0.543
$f_0 \sim f_9$		0.756	0.372	0.650	0.538

5.3.2 Comparison with Other DST Approaches

The previous subsection investigates how to get the RPN with the best configuration. In this subsection, the performance of RPN is compared to both rule-based and statistical approaches. To make fair comparison, all statistical models together with RPN in this subsection use similar feature set. Altogether, 2 rule-based trackers *MaxConf*, *HWU* and 2 statistical trackers *DNN*, *MaxEnt* which have been described in section 4.2.2 were built for performance comparison.

It can be observed that, with similar feature set, RPN can outperform both rule-based and statistical approaches in terms of joint goal accuracy. Statistical significance tests were also performed assuming a binomial distribution for each turn. RPN was shown to significantly outperform both rule-based and statistical approaches at 95% confidence level. For L2, RPN is competitive to both rule-based and the statistical approaches.

Table 5.3 Performance comparison among RPN, rule-based and statistical approaches with similar feature set on `dstc2eval` and `dstc3eval`. The performance of CMBP in the table is the performance of the RPN which has been initialized but not been trained.

Type	System	`dstc2eval`		`dstc3eval`	
		`Acc`	`L2`	`Acc`	`L2`
Rule	MaxConf	0.668	0.647	0.548	0.861
	HWU	0.720	0.445	0.594	0.570
Statistical	DNN	0.719	0.469	0.628	0.556
	MaxEnt	0.710	0.431	0.607	0.563
Mixed	CMBP	0.756	0.370	0.628	0.546
	RPN	0.757	0.372	0.645	0.543

5.3.3 Comparison with State-of-the-art DSTC Trackers

In the DSTCs, the state-of-the-art trackers mostly employed statistical approaches. Usually, richer feature set and more complicated model structures than the statistical models in section 5.3.2 are used. In this section, the proposed RPN approach is compared to the best submitted trackers in DSTC-2/3 and the best CMBP trackers, regardless of fairness of feature selection and the SLU refinement approach. RPN is compared and the results are shown in table 5.4 and table 5.5. Note that structure shown in figure 5.6 with richer feature set and a new recurrent connection is used here.

Table 5.4 Performance comparison among RPN, real-coefficient CMBP and best trackers of DSTC-2 on `dstc2eval`. Baseline* is the best results from the 4 baselines in DSTC2.

System	Approach	Rank	Acc	L2
Baseline*	Rule	5	0.719	0.464
Williams (2014)	LambdaMART	1	0.784	0.735
Henderson et al. (2014)	RNN	2	0.768	0.346
Sun et al. (2014)	DNN	3	0.750	0.416
Yu et al. (2015)	Real CMBP	2.5	0.762	0.436
RPN	RPN	2.5	0.756	0.372

Note that, in DSTC-2, the Williams (2014)'s system employed batch ASR hypothesis information (i.e. off-line ASR re-decoded results) and cannot be used in the normal on-line model in practice. Hence, the practically best tracker is Henderson et al. (2014).

It can be observed from table 5.4, RPN ranks only second to the best practical tracker in accuracy and L2. Considering that RPN only used probabilistic features and very limited added features and can operate very efficiently, it is quite competitive.

Table 5.5 Performance comparison among RPN, real-coefficient CMBP and best trackers of DSTC-3 on `dstc3eval`**. Baseline* is the best results from the 4 baselines in DSTC3.**

System	Approach	Rank	Acc	L2
Baseline*	Rule	6	0.575	0.691
Henderson et al. (2014)	RNN	1	0.646	0.538
Kadlec et al. (2014)	Rule	2	0.630	0.627
Sun et al. (2014)	Int CMBP	3	0.610	0.556
Yu et al. (2015)	Real CMBP	1.5	0.634	0.579
RPN	RPN	0.5	0.650	0.538

It can be seen from table 5.5, RPN trained on DSTC-2 can achieve state-of-the-art performance on DSTC-3 without modifying tracking method[1], outperforming all the submitted trackers in DSTC-3 including the RNN system. This demonstrates that RPN successfully inherits the advantage of good generalization ability of rule-based model.

[1] The parser is refined for DSTC-3 ([25], Zhu et al., 2014: 336–341.).

Chapter 6 Conclusion

6.1 Main Contribution

In this paper, two novel frameworks, *constrained Markov Bayesian polynomial* and *Recurrent Polynomial Network*, are proposed which manage to take advantage of both rule-based and statistical approaches. Both frameworks achieve efficiency, portability, interpretability and simplicity.

Constrained Markov Bayesian Polynomial framework takes the first step towards bridging the gap between rule-based and statistical approaches for dialogue state tracking. It uses a polynomial function to describe the probability operation rules and employs constraints to incorporate prior knowledge. By approximating descriptive constraints using linear constraints, the CMBP training is formulated as a standard problem of optimisation with linear constraints. Furthermore, the integer coefficient CMBP is extended to real-coefficient CMBP. With the ability of incorporating prior knowledge and being data-driven, CMBP has the advantages of both rule-based and statistical approaches.

The Recurrent Polynomial Network framework further bridges the gap between rule-based and statistical approaches for dialogue state tracking. With the ability of incorporating prior knowledge into a statistical framework, RPN has the advantages of both rule-based and statistical approaches.

Experiments on two dialog state tracking challenge (DSTC) tasks showed that both frameworks not only are more stable than many major statistical approaches, but also have competitive performance, outperforming many state-of-the-art trackers.

6.2 Future Work

It is interesting to note that the second best tracker in DSTC-3 was a rule-based tracker. Different from the general form in equation (4-4), it further conditioned the Bayesian polynomial on the system act of the previous turn. This can be regarded as a piece-wise polynomial extension of equation (4-4). Since both the CMBP and RPN frameworks are easy to extend, future work will investigate piece-wise polynomials.

Furthermore, tracking dialogue states for sub-dialogue segments in human-human dialogues will be the focus of the next Dialog State Tracking Challenge (i.e. DSTC-4). It is also interesting to investigate applying CMBP and RPN to that task.

REFERENCE

[1] WILLIAMS J D, YOUNG S. Partially observable Markov decision processes for spoken dialog systems[J]. Computer Speech & Language, 2007, 21(2):393–422.

[2] THOMSON B, YOUNG S. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems[J]. Computer Speech & Language, 2010, 24(4):562–588.

[3] YOUNG S, GAŠIĆ M, KEIZER S, et al. The hidden information state model: A practical framework for POMDP-based spoken dialogue management[J]. Computer Speech & Language, 2010, 24(2):150–174.

[4] YOUNG S, GASIC M, THOMSON B, et al. POMDP-Based Statistical Spoken Dialog Systems: A Review[J]. Proceedings of the IEEE, 2013, 101(5):1160–1179.

[5] WILLIAMS J D. Challenges and opportunities for state tracking in statistical spoken dialog systems: Results from two public deployments[J]. Selected Topics in Signal Processing, IEEE Journal of, 2012, 6(8):959–970.

[6] WILLIAMS J D. A belief tracking challenge task for spoken dialog systems[C]//NAACL-HLT Workshop on Future Directions and Needs in the Spoken Dialog Community: Tools and Data. .[S.l.]: [s.n.] , 2012:23–24.

[7] WILLIAMS J, RAUX A, RAMACHANDRAN D, et al. The Dialog State Tracking Challenge[C]//Proceedings of the SIGDIAL 2013 Conference. Metz, France: Association for Computational Linguistics, 2013:404–413. http://www.aclweb.org/anthology/W/W13/W13-4065.

[8] HENDERSON M, THOMSON B, WILLIAMS J D. The Second Dialog State Tracking Challenge[C]//Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL). Philadelphia, PA, U.S.A.: Association for Computational Linguistics, 2014:263–272. http://www.aclweb.org/anthology/W14-4337.

[9] HENDERSON M, THOMSON B, WILLIAMS J D. The Third Dialog State Tracking Challenge[C]//Proceedings of IEEE Spoken Language Technology Workshop (SLT). .[S.l.]: [s.n.] , 2014:324–329.

[10] SUN K, CHEN L, ZHU S, et al. A GENERALIZED RULE BASED TRACKER FOR DIALOGUE STATE TRACKING[C]//Proceedings of IEEE Spoken Language Technology Workshop (SLT). .[S.l.]: [s.n.] , 2014:330–335.

[11] YU K, SUN K, CHEN L, et al. Constrained Markov Bayesian Polynomial for Efficient Dialogue State Tracking[J]. submitted to IEEE Transactions on Audio, Speech and Language Processing, 2015.

[12] SUN K, XIE Q, YU K. Recurrent Polynomial Network for Dialogue State Tracking[J]. submitted to Dialogue and Discourse, 2015.

[13] LEE S, ESKENAZI M. Recipe For Building Robust Spoken Dialog State Trackers: Dialog State Tracking Challenge System Description[C]//Proceedings of the SIGDIAL 2013 Conference. Metz, France: Association for Computational Linguistics, 2013:414–422. http://www.aclweb.org/anthology/W/W13/W13-4066.

[14] SUN K, CHEN L, ZHU S, et al. The SJTU System for Dialog State Tracking Challenge 2[C]//Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL). Philadelphia, PA, U.S.A.: Association for Computational Linguistics, 2014:318–326. http://www.aclweb.org/anthology/W14-4343.

[15] LEE S. Structured Discriminative Model For Dialog State Tracking[C]//Proceedings of the SIGDIAL 2013 Conference. Metz, France: Association for Computational Linguistics, 2013:442–451. http://www.aclweb.org/anthology/W/W13/W13-4069.

[16] KIM S, BANCHS R. Sequential Labeling for Tracking Dynamic Dialog States[C]//Proceedings of the SIGDIAL Conference. .[S.l.]: [s.n.] , 2014:332–336.

[17] HENDERSON M, THOMSON B, YOUNG S. Deep Neural Network Approach for the Dialog State Tracking Challenge[C]//Proceedings of the SIGDIAL 2013 Conference. Metz, France: Association for Computational Linguistics, 2013:467–471. http://www.aclweb.org/anthology/W/W13/W13-4073.

[18] HENDERSON M, THOMSON B, YOUNG S. Word-Based Dialog State Tracking with Recurrent Neural Networks[C]//Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL). Philadelphia,

PA, U.S.A.: Association for Computational Linguistics, 2014:292–299. http://www.aclweb.org/anthology/W14-4340.

[19] WILLIAMS J D. Web-style ranking and SLU combination for dialog state tracking[C]//Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL). Philadelphia, PA, U.S.A.: Association for Computational Linguistics, 2014:282–291. http://www.aclweb.org/anthology/W14-4339.

[20] ZILKA L, MAREK D, KORVAS M, et al. Comparison of Bayesian Discriminative and Generative Models for Dialogue State Tracking[C]//Proceedings of the SIGDIAL 2013 Conference. Metz, France: Association for Computational Linguistics, 2013:452–456. http://www.aclweb.org/anthology/W/W13/W13-4070.

[21] WANG Z, LEMON O. A Simple and Generic Belief Tracking Mechanism for the Dialog State Tracking Challenge: On the believability of observed information[C]//Proceedings of the SIGDIAL 2013 Conference. Metz, France: Association for Computational Linguistics, 2013:423–432. http://www.aclweb.org/anthology/W/W13/W13-4067.

[22] KADLEC R, VODOLN M, LIBOVICK J, et al. KNOWLEDGE-BASED DIALOG STATE TRACKING[C]//Proceedings 2014 IEEE Spoken Language Technology Workshop. South Lake Tahoe, USA: [s.n.] , 2014:348–353.

[23] ACHTERBERG T. SCIP: Solving constraint integer programs[J]. Mathematical Programming Computation, 2009, 1(1):1–41.

[24] HENDERSON M, THOMSON B, WILLIAMS J. Dialog State Tracking Challenge 2 & 3[J]. 2013.

[25] ZHU S, CHEN L, SUN K, et al. SEMANTIC PARSER ENHANCEMENT FOR DIALOGUE DOMAIN EXTENSION WITH LITTLE DATA[C]//Proceedings of IEEE Spoken Language Technology Workshop (SLT). .[S.l.]: [s.n.] , 2014:336–341.

[26] HENDERSON M, THOMSON B, YOUNG S. Robust Dialog State Tracking Using Delexicalised Recurrent Neural Networks and Unsupervised Adaptation[C]//Proceedings of IEEE Spoken Language Technology Workshop (SLT). .[S.l.]: [s.n.] , 2014:360–365.

[27] WANG Z. HWU Baseline Belief Tracker for DSTC 2 & 3[R].[S.l.]: [s.n.] , 2013. http://camdial.org/~mh521/dstc/downloads/HWU_baseline.zip.

[28] XIE Q, SUN K, ZHU S, et al. Recurrent Polynomial Network for Dialogue State Tracking with Mismatched Semantic Parsers[J]. submitted to the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), 2015.

Appendix

CMBP Constraints Formulation

In order to be consistent with section 4.1.2 and introduce the constraints clearly, the constraints formulation of order-3 CMBP is the focus in the following content. The constraints formulation of CMP of other order can be obtained with just slight modifications of the constraints formulation of order-3 CMBP. As definition (4-2), the coefficients of CMBP of order 3 is denoted by w_{ijk}:

$$\mathcal{P}(x_1, x_2, x_3, x_4, x_5, x_6) = \sum_{0 \leq i \leq j \leq k \leq 6} w_{ijk} x_i x_j x_k \tag{6-1}$$

where $x_0 = 1$, and $w \in \mathbb{Z}$.

Constraints Formulation

The probabilistic constraints, intuition constraints, and regularization constrains investigated in this paper are described below respectively.

Probabilistic constraints:

$$0 \leq b_t(v) \leq 1 \tag{6-2}$$

$$0 \leq b_t^r \leq 1 \tag{6-3}$$

$$b_t^r = 1 - \sum_{v'} b_t(v') \tag{6-4}$$

Intuition constraints:

- If neither positive nor negative information is collected, the belief should not change.

$$\begin{aligned} &P^+_{t+1}(v) = 0 \wedge P^-_{t+1}(v) = 0 \wedge \tilde{P}^+_{t+1}(v) = 0 \wedge \\ &\tilde{P}^-_{t+1}(v) = 0 \quad \Rightarrow \quad b_{t+1}(v) = b_t(v) \end{aligned} \tag{6-5}$$

where here "$\wedge$" and "$\Rightarrow$" are used to denote logical conjunction and material implication respectively.

- If both ASR and SLU is perfectly correct, that is, 1 is assigned to all correct values and 0 to all incorrect values, then the model should always give the correct result. Considering the special case that there is only one value which is not "`None`", the following 3 constraints can be obtained.

$$P^+_t(v) = 1 \Rightarrow b_t(v) \geq 0.5 \tag{6-6}$$

$$P^-_t(v) = 1 \Rightarrow b_t(v) \leq 0.5 \tag{6-7}$$

$$\tilde{P}^+_t(v) = 1 \Rightarrow b_t(v) \leq 0.5 \tag{6-8}$$

- The belief should be unchanged or positively correlated with the positive scores from SLU.

$$\frac{\partial \mathcal{P}(P^+_{t+1}(v), P^-_{t+1}(v), \tilde{P}^+_{t+1}(v), \tilde{P}^-_{t+1}(v), b^r_t, b_t(v))}{\partial P^+_{t+1}(v)} \geq 0 \tag{6-9}$$

- The belief should be unchanged or negatively correlated with the negative scores from SLU.

$$\frac{\partial \mathcal{P}(P^+_{t+1}(v), P^-_{t+1}(v), \tilde{P}^+_{t+1}(v), \tilde{P}^-_{t+1}(v), b^r_t, b_t(v))}{\partial P^-_{t+1}(v)} \leq 0 \tag{6-10}$$

- The belief should be unchanged or negatively correlated with the sum of the positive scores of the other values.

$$\frac{\partial \mathcal{P}(P^+_{t+1}(v), P^-_{t+1}(v), \tilde{P}^+_{t+1}(v), \tilde{P}^-_{t+1}(v), b^r_t, b_t(v))}{\partial \tilde{P}^+_{t+1}(v)} \leq 0 \tag{6-11}$$

- The belief should be unchanged or positively correlated with the sum of the negative scores of the other values.

$$\frac{\partial \mathcal{P}(P^+_{t+1}(v), P^-_{t+1}(v), \tilde{P}^+_{t+1}(v), \tilde{P}^-_{t+1}(v), b^r_t, b_t(v))}{\partial \tilde{P}^-_{t+1}(v)} \geq 0 \tag{6-12}$$

- The belief of the current turn should be unchanged or positively correlated with the belief of the previous turn.

$$\frac{\partial \mathcal{P}(P^+_{t+1}(v), P^-_{t+1}(v), \tilde{P}^+_{t+1}(v), \tilde{P}^-_{t+1}(v), b^r_t, b_t(v))}{\partial b_t(v)} \geq 0 \tag{6-13}$$

Regularization constraints:

- The coefficients of $\mathcal{P}(\cdot)$ is limited to be in $[-1, 1]$. This constraint comes from the observation that all coefficients of rule-based model (3-3) are in $[-1, 1]$.

$$-1 \leq w_{ijk} \leq 1 \tag{6-14}$$

- The sum of the coefficients of $\mathcal{P}(\cdot)$ is limited to be 0. This constraint comes from the observation that the sum of the coefficients of rule-based model (3-3) is 0.

$$\sum_{0 \leq i \leq j \leq k \leq 6} w_{ijk} = 0 \tag{6-15}$$

Constraints Approximation

To simplify the presentation, the set consisting of all possible input vectors $(P^+_t(v)$, $P^-_t(v)$, $\tilde{P}^+_t(v)$, $\tilde{P}^-_t(v)$, $b^r_t, b_t(v))$ is denoted by χ. By definition, the following relations and (6-2), (6-3), (6-4) are true:

$$0 \leq P_t^+(v) \leq 1 \tag{6-16}$$

$$0 \leq P_t^-(v) \leq 1 \tag{6-17}$$

$$0 \leq \tilde{P}_t^+(v) \leq 1 \tag{6-18}$$

$$0 \leq \tilde{P}_t^-(v) \leq 1 \tag{6-19}$$

$$0 \leq P_t^+(v) + \tilde{P}_t^+(v) \leq 1 \tag{6-20}$$

$$0 \leq P_t^-(v) + \tilde{P}_t^-(v) \leq 1 \tag{6-21}$$

Therefore,

$$\begin{aligned} \boldsymbol{\chi} = \{(x_1, x_2, x_3, x_4, x_5, x_6) | 0 \leq x_1 \leq 1 \wedge 0 \leq x_2 \leq 1 \wedge \\ 0 \leq x_3 \leq 1 \wedge 0 \leq x_4 \leq 1 \wedge x_1 + x_3 \leq 1 \wedge x_2 + x_4 \leq 1 \wedge \\ 0 \leq x_5 \leq 1 \wedge 0 \leq x_6 \leq 1 \wedge x_5 + x_6 \leq 1\} \end{aligned} \tag{6-22}$$

The conversion from the exact constraints to the relaxed linear constraints is discussed in detail as below. For approximation purpose, two quantised interval of $[0, 1]$, T_5 and T_{10}, need to be defined first:

$$\begin{aligned} T_5 &= \{0, 0.2, 0.4, 0.6, 0.8, 1\} \\ T_{10} &= \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\} \end{aligned}$$

A number of theorems are then proved for the constraints approximation.

Theorem 6.1. If a rule satisfies constraints (6-2), (6-3), (6-4), then the rule satisfies the following sets of linear constraints:

$$\{0 \leq \mathcal{P}(\boldsymbol{a}) \leq 1 | \boldsymbol{a} \in \boldsymbol{\chi}, a_i \in T_5, i = 1, \cdots, 6\} \tag{6-23}$$

$$\begin{aligned}
&\{0 \leq \mathcal{P}(\boldsymbol{a}) + \mathcal{P}(\boldsymbol{b}) \leq 1 | \boldsymbol{a}, \boldsymbol{b} \in \boldsymbol{\chi}, a_1 + a_3 = b_1 + b_3, \\
&a_2 + a_4 = b_2 + b_4, a_1 \leq b_3, b_1 \leq a_3, a_2 \leq b_4, b_2 \leq a_4, \\
&a_5 = b_5, a_5 + a_6 + b_6 = 1, a_i, b_i \in T_5, i = 1, \cdots, 6\}
\end{aligned} \tag{6-24}$$

Proof. The set of linear constraints (6-23) can be obtained by constraint (6-2). By combining constraint (6-3) and (6-4), it can be proved that $0 \leq \sum_{v'} b_t(v') \leq 1$. Thus the set of linear constraints (6-24) can be obtained by considering the special case that there are at least 2 values which are not "`None`". □

Theorem 6.2. A rule satisfies constraint (6-5) if and only if

$$\begin{aligned}
&w_{000} = w_{005} = w_{055} = w_{056} = w_{066} = \\
&w_{555} = w_{556} = w_{566} = w_{666} = 0
\end{aligned} \tag{6-25}$$

and

$$w_{006} = 1 \tag{6-26}$$

Proof. Suppose constraints (6-25) and (6-26) hold. Under the condition $P^+_{t+1}(v) = P^-_{t+1}(v) = \tilde{P}^+_{t+1}(v) = \tilde{P}^-_{t+1}(v) = 0$, then for all v, $(P^+_{t+1}(v) = P^-_{t+1}(v) = 0)$ by the definition of $P^+_{t+1}(v)$, $P^-_{t+1}(v)$, $\tilde{P}^+_{t+1}(v)$, $\tilde{P}^-_{t+1}(v)$ and constraints (6-16), (6-17). Thus by definition (4-4) and equation (6-1)

$$\begin{aligned}
b_{t+1}(v) &= w_{000} + w_{005} b^r_t + w_{055}(b^r_t)^2 + w_{056} b^r_t b_t(v) \\
&+ w_{066}(b_t(v))^2 + w_{555}(b^r_t)^3 + w_{556}(b^r_t)^2 b_t(v) \\
&+ w_{566} b^r_t (b_t(v))^2 + w_{666}(b_t(v))^3 + w_{006} b_t(v) \\
&= b_t(v)
\end{aligned}$$

Therefore, constraint (6-5) holds. Reversely suppose constraint (6-5) holds, it is easy to check that under the condition that $P^+_{t+1}(v) = 0 \wedge P^-_{t+1}(v) = 0 \wedge \tilde{P}^+_{t+1}(v) = 0 \wedge \tilde{P}^-_{t+1}(v) = 0$, if at least one of constraint (6-25) or (6-26) does not hold, the equality "$b_{t+1}(v) \equiv b_t(v)$" does not hold. □

Theorem 6.3. If a rule satisfies constraints (6-6), (6-7), (6-8), then the rule satisfies the following set of linear constraints:

$$\{\mathcal{P}(1,0,0,0,a_5,0) \geq 0.5 | a_5 \in T_{10}\} \tag{6-27}$$

$$\{\mathcal{P}(0,1,0,0,a_5,a_6) \leq 0.5 | a_5, a_6 \in T_{10}, a_5 + a_6 = 1\} \tag{6-28}$$

$$\{\mathcal{P}(0,0,1,0,a_5,a_6) \leq 0.5 | a_5, a_6 \in T_{10}, a_5 + a_6 = 1\} \tag{6-29}$$

Proof. The set of linear constraints (6-27) can be obtained by simply combining constraint (6-6) and definition (4-4). The derivations for the sets of linear constraints (6-28) and (6-29) are similar. □

Theorem 6.4. If a rule satisfies constraints (6-9), (6-10), (6-11), (6-12), (6-13), then the rule satisfies the following sets of linear constraints:

$$\left\{ \mathcal{P}(\mathbf{a}) \geq \mathcal{P}(\mathbf{b}) \;\middle|\; \begin{array}{l} \mathbf{a}, \mathbf{b} \in \boldsymbol{\chi}, a_i \in T_5, i = 1, \cdots, 6 \\ a_1 = b_1 + 0.1, \quad a_i = b_i \quad \forall i \neq 1 \end{array} \right\} \tag{6-30}$$

$$\left\{ \mathcal{P}(\mathbf{a}) \geq \mathcal{P}(\mathbf{b}) \;\middle|\; \begin{array}{l} \mathbf{a}, \mathbf{b} \in \boldsymbol{\chi}, a_i \in T_5, i = 1, \cdots, 6 \\ a_2 = b_2 + 0.1, \quad a_i = b_i \quad \forall i \neq 2 \end{array} \right\} \tag{6-31}$$

$$\left\{ \mathcal{P}(\mathbf{a}) \geq \mathcal{P}(\mathbf{b}) \;\middle|\; \begin{array}{l} \mathbf{a}, \mathbf{b} \in \boldsymbol{\chi}, a_i \in T_5, i = 1, \cdots, 6 \\ a_3 = b_3 + 0.1, \quad a_i = b_i \quad \forall i \neq 3 \end{array} \right\} \tag{6-32}$$

$$\left\{ \mathcal{P}(\mathbf{a}) \geq \mathcal{P}(\mathbf{b}) \;\middle|\; \begin{array}{l} \mathbf{a}, \mathbf{b} \in \boldsymbol{\chi}, a_i \in T_5, i = 1, \cdots, 6 \\ a_4 = b_4 + 0.1, \quad a_i = b_i \quad \forall i \neq 4 \end{array} \right\} \tag{6-33}$$

$$\left\{ \mathcal{P}(\mathbf{a}) \geq \mathcal{P}(\mathbf{b}) \;\middle|\; \begin{array}{l} \mathbf{a}, \mathbf{b} \in \boldsymbol{\chi}, a_i \in T_5, i = 1, \cdots, 6 \\ a_6 = b_6 + 0.1, \quad a_i = b_i \quad \forall i \neq 6 \end{array} \right\} \tag{6-34}$$

Proof. The rule satisfies the set of linear constraints (6-30) is because constraint (6-9) indicates $\mathcal{P}(x_1, x_2, x_3, x_4, x_5, x_6)$ is monotonically increasing with respect to x_1. The derivations for the other sets of linear constraints are similar. □

By theorem 6.1, 6.2, 6.3 and 6.4, it can be seen that the linear constraints (6-23) - (6-34) relax the constraints (6-2) - (6-4), (6-5) - (6-13).

Derivative calculation

Using MSE as the criterion, $\delta_x^{(t)} = \frac{\partial \mathcal{L}}{\partial u_x^{(t)}}$ is initialized as following:

$$\delta_x^{(t)} = \begin{cases} 2(u_x^{(t)} - l_t) & \text{if } x \text{ is a output node} \\ 0 & \text{otherwise} \end{cases} \tag{6-35}$$

Suppose node x is an activation node and $f(\cdot) = softclip(\cdot)$, let $y = o_x$,

$$\begin{aligned} \delta_y^{(t)} &= \frac{\partial \mathcal{L}}{\partial u_y^{(t)}} \\ &= \frac{\partial \mathcal{L}}{\partial u_x^{(t)}} \frac{\partial u_x^{(t)}}{\partial u_y^{(t)}} \\ &= \delta_x^{(t)} \frac{\partial f(u_y^{(t)})}{\partial u_y^{(t)}} \end{aligned} \tag{6-36}$$

Suppose node $x = (d, i)$ is a sum node, then when node x passes its error, the error of node $y \in \hat{I}_x$ is updated as

$$\begin{aligned} \delta_y^{(t)} &= \delta_y^{(t)} + \frac{\partial \mathcal{L}}{\partial u_x^{(t)}} \frac{\partial u_x^{(t)}}{\partial u_y^{(t)}} \\ &= \delta_y^{(t)} + \delta_x^{(t)} \hat{w}_{x,y} \end{aligned} \tag{6-37}$$

Similarly, error of node $y \in I_x$ is updated as

$$\begin{aligned} \delta_y^{(t)} &= \delta_y^{(t)} + \frac{\partial \mathcal{L}}{\partial u_x^{(t)}} \frac{\partial u_x^{(t)}}{\partial u_y^{(t-1)}} \\ &= \delta_y^{(t)} + \delta_x^{(t)} w_{x,y} \end{aligned} \tag{6-38}$$

Suppose node $x = (d, i)$ is a product node, then when node x passes its error, error of node $y \in \hat{I}_x$ is updated as

$$\begin{aligned}\delta_y^{(t)} &= \delta_y^{(t)} + \frac{\partial \mathcal{L}}{\partial u_x^{(t)}} \frac{\partial u_x^{(t)}}{\partial u_y^{(t)}} \\ &= \delta_y^{(t)} + \delta_x^{(t)} \hat{M}_{x,y} {u_y^{(t)}}^{\hat{M}_{x,y}-1} \prod_{z \in \hat{I}_x - \{y\}} {u_z^{(t)}}^{\hat{M}_{x,z}} \prod_{z \in I_x} {u_z^{(t-1)}}^{M_{x,z}}\end{aligned} \tag{6-39}$$

Similarly, error of node $y \in I_x$ is updated as

$$\begin{aligned}\delta_y^{(t)} &= \delta_y^{(t)} + \frac{\partial \mathcal{L}}{\partial u_x^{(t)}} \frac{\partial u_x^{(t)}}{\partial u_y^{(t-1)}} \\ &= \delta_y^{(t)} + \delta_x^{(t)} M_{x,y} {u_y^{(t-1)}}^{M_{x,y}-1} \prod_{z \in \hat{I}_x} {u_z^{(t)}}^{\hat{M}_{x,z}} \prod_{z \in I_x - \{y\}} {u_z^{(t-1)}}^{M_{x,z}}\end{aligned} \tag{6-40}$$

Undergraduate Publications

1. **Kai Sun**, Lu Chen, Su Zhu and Kai Yu. The SJTU System for Dialog State Tracking Challenge 2. The 15th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL), Pennsylvania, USA, 2014: 318-326.

2. **Kai Sun**, Lu Chen, Su Zhu and Kai Yu. A Generalized Rule Based Tracker for Dialogue State Tracking. IEEE Spoken Language Technology Workshop (SLT), South Lake Tahoe, USA, 2014: 330-335.

3. Qizhe Xie, **Kai Sun**, Su Zhu, Lu Chen and Kai Yu. Recurrent Polynomial Network for Dialogue State Tracking with Mismatched Semantic Parsers. To appear in the 16th Annual SIGdial Meeting on Discourse and Dialogue (SIGdial). 2015.

4. Su Zhu, Lu Chen, **Kai Sun**, Da Zheng and Kai Yu. Semantic Parser Enhancement for Dialogue Domain Extension with Little Data. IEEE Spoken Language Technology Workshop (SLT), South Lake Tahoe, USA, 2014: 336-341.

5. Kai Yu, Lu Chen, Bo Chen, **Kai Sun** and Su Zhu. Cognitive Technology in Task-oriented Dialogue Systems Concepts, Advances and Future. Chinese Journal of Computers, 2015, 37.

6. **Kai Sun**, Qizhe Xie and Kai Yu. Recurrent Polynomial Network for Dialogue State Tracking. Submitted to Dialogue and Discourse (D&D), 2015.

7. Kai Yu, **Kai Sun**, Lu Chen and Su Zhu. Constrained Markov Bayesian Polynomial for Efficient Dialogue State Tracking. Submitted to IEEE Transactions on Audio, Speech and Language Processing (TASL). 2015.

8. Kai Yu, Lu Chen, **Kai Sun**, Su Zhu, Qizhe Xie. Evolvable Dialogue State Tracking for Statistical Dialogue Management. Submitted to Frontiers of Computer Science. 2015

Acknowledgements

I would like to express my deepest gratitude to my advisor Prof. Kai Yu, who has always been there to listen and give practical advice. His patience, enthusiasm, great kindness and immense knowledge helped me in all the time of research. I could not have asked for a better advisor.

I am also thankful to SJTU Speech Lab for providing wonderful academic facilities and atmosphere. In particular, many thanks to my friends Lu Chen, Su Zhu, Qizhe Xie, and Da Zheng in Speech Lab for numerous discussions on related topics and collaborations.

This work was supported by the Program for Professor of Special Appointment (Eastern Scholar) at Shanghai Institutions of Higher Learning and the China NSFC project No. 61222208. The author was with the support of the Chun-Tsung Scholarship.