Associative Knowledge Feature Vector Inferred on External Knowledge Base for Dialog State Tracking

Yukitoshi Murase[a], Yoshino Koichiro[a,b,c], Satoshi Nakamura[a]

[a]*Nara Institute of Science and Technology, Takayamacho 8916-5, Ikoma, Nara, 630-0192, Japan*
[b]*RIKEN AIP, Takayamacho 8916-5, Ikoma, Nara, 630-0192, Japan*
[c]*JST PRESTO, Honcho 4-1-8, Kawaguchi, Saitama, 332-0012, Japan*

Abstract

The dialog state tracker is one of the most important modules on task-oriented dialog systems, its accuracy strongly affects the quality of the system response. The architecture of the tracker has been changed from pipeline processing to an end-to-end approach that directly estimates a user's intention from each current utterance and a dialog history because of the growth in the use of the neural-network-based classifier. However, tracking appropriate slot-value pairs of dialog states that are not explicitly mentioned in user utterances is still a difficult problem. In this research, we propose creating feature vectors by using inference results on an external knowledge base. This inference process predicts associative entities in the knowledge base, which contribute to the dialog state tracker for unseen entities of utterances. We extracted a part of a graph structure from an external knowledge base (Wikidata). Label propagation was used for inferring associative nodes (entities) on the graph structure to produce feature vectors. We used the vectors for the input of a fully connected neural network (FCNN) based tracker. We also introduce a convolutional neural network (CNN) tracker as a state-of-the-art tracker and ensemble models of FCNN and CNN trackers. We used a common test bed, *Dialog State Tracking Challenge 4* for experiments. We confirmed the effectiveness of the associative knowledge feature vector, and one ensemble model outperformed other models.

Keywords: Dialog State Tracking, Knowledge Base, Knowledge Graph, Associative Knowledge Inference

---

1. Introduction

Dialog state tracking (DST) is known as an important component of task-oriented dialog systems [1, 2, 3]. DST is a task of tracking user intentions (dialog frame) from input utterances and dialog history. Dialog state tracking challenges (DSTCs) have been held to provide a common test bed for DST [4]. DSTC, DSTC2, and DSTC3 provide human-computer dialog corpora for estimating a user's dialog states [4, 5, 6]. DSTC4 and DSTC5 provide human-human dialog corpora for estimating the states. The difficulties of estimating human-human conversation are the large intentional space and collecting enough data for covering the space.

Throughout the previous DSTCs, discriminative methods, which directly predict dialog frames, performed well [7]. Recurrent neural network (RNN) based approaches competitively performed well on DSTC2 [8]. RNN approaches were also competitive to other approaches for DSTC4 and 5 [9, 10]; however, convolutional neural network (CNN) based approaches outperformed RNN-based approaches and were reported as being the state-of-the-art for DSTC5 [11, 12]. These approaches used distributed word representation such as word2vec or GloVe [13, 14] for their input features. However, the lack of information obtained from input features is still a problem from two viewpoints. The first is that it is challenging to find any unseen slot-values of dialog state frames that are not observed in a user utterance. The second is data size; the number of annotated dialog data is limited, and the number of output states is explosively large.

External knowledge such as ontology is a pivotal component for solving the problem of a lack of information in inputs. However, handcrafted ontologies are not capable of being extended without professionals. Due to the growth of the world wide web (WWW), a variety of knowledge bases (KBs) is publicly available [15, 16]. In Ma et al. [17], a method of ontology extension was proposed uses external KBs. These KBs contain entities and properties that are transformed into a graphical model. It is possible to find associative entities by using the structure of KBs to fill a lack of input information.

In this paper, we propose a method of creating an associative knowledge feature vectors (AKFVs) highly capable of expressing the meaning of an utterance by using unobservable information in utterances. The feature vectors include information obtained from global associative entities. A fully connected neural network (FCNN) with the proposed feature vectors comparably performed the state-of-the-art CNN-based dialog state tracker. Moreover, an ensemble of the proposed method and the CNN-based tracker outperformed the CNN-based tracker and achieved the best score for neural-network-based trackers for DSTC4.

2. Related Works

2.1. Dialog State Tracking Challenge 4

Dialog State Tracking Challenge 4 (DSTC4) is a common test bed for dialog state tracking and is aimed at achieving more human-like dialog systems by using a human-human conversation corpus for a sightseeing domain. The corpus is a collection that a total of 21 hours of conversation between 3 tour guides and 35 tourists on Skype. It is divided into *training*, *development*, and *test set*, which respectively contain 14, 6, and 15 dialog sessions. Each dialog session has been manually transcribed, and dialog frames have been annotated for each sub-dialog level. A sub-dialog means any turns of the dialog session.

The total number of utterances within sub-dialog segments is 20,641. The challenge of DSTC4 is a task of tracking dialog frames for sub-dialog segments, where a frame contains a topic and slot-value pairs. An ontology is also given data, and it contains all possible slot-value pairs under each topic. The slot-values represents intention in the human conversation, and the ontology indicates the knowledge of possible human intentions for this domain.

Annotations are given to each sub-dialog segment. The topics are subdomains under the sightseeing domain, and topics are annotated to the all sub-dialog segments. Topics are categorized into five classes: *accommodation*, *attraction*, *food*, *shopping*, and *transportation*. A state frame is given for all sub-dialog segments. The frames contain several slot-value pairs, which represent intentions in conversation within each sub-dialog. For example, the slot-value annotation at the *accommodation* topic frame might have a slot called "type" that represents *"What kind of accommodation style?"*, and corresponding values could be filled with *hotel*, *hostel*, etc.

The ontology is given to ensure estimation of slot-value pairs since it contains all topics and possible pairs in a hierarchical structure. The structure has three layers. The top layer has five topics, and the middle layer has multiple slots that are each dependent on a topic. The bottom layer has values, which depend on a slot. Therefore, a higher layer has more abstract information, and a lower layer has more specific information as shown in Table 1.

Table 1: Example of Ontology's Hierarchical Structure

Topic	**Slot**	**Value**
Accommodation	INFO	...
	Name	InnCrowd Backpackers Hostel
		...
	Type	Hotel
		Hostel
		...
	...	...
Food	INFO	...
	...	...
		...
...	INFO	...

The number of all slot-value pairs is 5,608, so the task requires estimation within a large intentional space.

2.2. Dialog State Tracker with External Knowledge Base

A related approach tried to estimate a user's goals (dialog states) by inference on a large-scale knowledge base (KB) instead of searching on look-up Tables [17]. This framework indicates the possibility of estimating unobserved states by using an inference method on an external KB. In other words, the inference method makes it possible to associate any observed words and unobservable entities. This approach transforms a KB into a graphical model, and a *Markov random field* (MRF) is applied to any inference method on the graph. This graph contains two different types of nodes: named-entity nodes and attribute nodes.

This method utilizes top-n results, the local results from inferring unobserved information on a knowledge graph. However, inferred results contain more associative knowledge information even if the inference scores are low. Instead, we try to use information, the combination of scores, to create more expressive feature vectors of utterances that represents global associative knowledge by using the whole knowledge space.

2.3. External Knowledge in Neural-network Approaches

The way to use external knowledge bases in neural-networks is widely researched. Using graph embedding and memory networks are efficient ways to

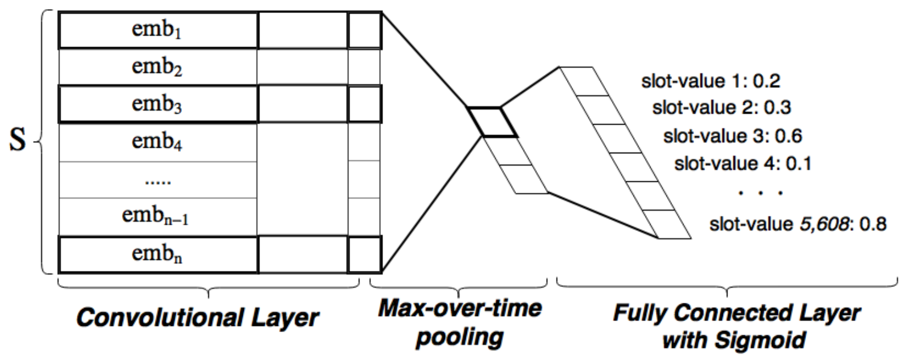

Figure 1: Overview of CNN model

carry information from knowledge bases [18, 19, 20]. Graph convolution can automatically infers related features from the knowledge graph. However, these approaches have a disadvantage on interpretability, it is hard to interpret the effective feature from the trained neural network model. In contrast, our inferring method on a knowledge graph based has an advantage to find how a external knowledge worked for the task.

2.4. Convolutional Neural Network-based Dialog State Trackers

The convolutional-neural-network-based tracker (CNN-based tracker) is the state-of-the-art dialog state tracker for DSTC5 [12]. In Shi et al. [11] a CNN-based tracker was also applied to estimate the *'INFO'* slot for DSTC4, and this model performed well within neural network approaches. These approaches are based on CNN sentence classification models [21, 22] that can consider the meanings of words and word orders in an utterance.

The CNN classifier lists embedding vectors of words in an utterance as shown in “Convolutional Layer” in Figure 1. Word vectors are drawn up with the order in the sentence to construct a matrix. The pre-trained word embedding model is used to convert words into embedding vectors according to the distributional hypothesis. Local features are extracted by the filter of a convolutional layer, and thus, local connections of words in an utterance can be captured in this architecture to express the meaning of the utterance.

3. Associative Knowledge Feature Inference on Knowledge Graph

Our approach is creating an associative knowledge feature vector from the external knowledge, and the vector is represented by inference on a knowledge graph. This approach takes the observed words (entities) for an utterance as the input and produces a feature vector that contains information on associative entities. The produced feature vector contains whole information that can be inferred on a knowledge graph, and thus, we can utilize the global associative knowledge information.

3.1. Graph Transformation of Knowledge Base

KBs contain entities (subjects and objects) and relations (predicates) such as the triplet form of subject-predicate-object. A knowledge graph can be created from these triplets by connecting the entities with relations. We extract these triplets from Wikidata, a knowledge base with multiple-languages, to create a graph. In this research, only English entities are used to create a graph. We remind the reader that Wikidata can be applied to any language. We use entities as nodes and relations as edges to transform a KB into a graph.

3.2. Subgraph Creation from Wikidata

The remaining problem is that Wikidata is too large for working with any inference method on a graph; thus, we selected a part of the entities and relations to utilize a KB for feature inference. We take a subgraph that suits the domain, that is, the DSTC4 training/development dataset. We used entities observed in the training set and other entities that have any relations to the observed entities. Additionally, NLTK stopwords are removed from the candidates of entities.

The named entities, which match observed words, are added as nodes of the subgraph. We call these nodes *core nodes*. Entities related to *core nodes* are also added to the subgraph. We call these nodes *neighboring nodes*. In addition, 1-hop away nodes that are related to *neighboring nodes* in Wikidata are added to the subgraph to enlarge the knowledge space. We use the defined relations in Wikidata to find the *neighboring nodes* and the *1-hop away nodes*. These additional nodes give more meaningful information through inference. Edges are added for all related entities to complete subgraph creation no matter what the kinds of relations.

An example of a subgraph creation is shown in Figure 2 and consists of core nodes of *"Singapore."* The *"Singapore"* node is added on the subgraph with its neighboring nodes (*"Asia"*, *"City"*, *"Island Nation"*, *"Country"* and *"Malaysia"*). Nodes with a 1-hop relation are also added: (*"Area"* and *"Continent"*). In addition, we assume that Malaysia is also observed

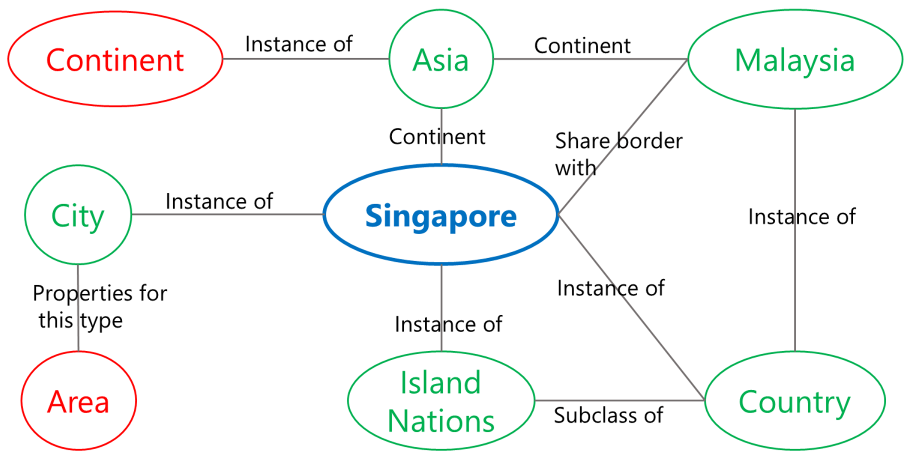

Figure 2: Example of graph creation: blue node is *core node*, green nodes are *neighboring nodes*, and red nodes are *1-hop away nodes*.

in utterances, and related nodes in Wikidata are connected to the nodes of *"Malaysia"* (*"Country"* and *"Asia"*).

3.3. Inference Method: Label Propagation on Subgraph

Label propagation is a method of updating values of nodes in a graph, given the values of each node by minimizing two objective functions: differences between a given value and updated value and differences between neighboring nodes. We exploit this method for representing the knowledge states that contain associative knowledge information. Our proposed method sets observed-class and unobserved-class labels from observations of utterances (observed=1, unobserved=0). The value of each node is updated by propagating observed class labels.

In the label propagation algorithm that we use, edges are represented as $\mathbf{W}$. $\mathbf{W}$ is an $N \times N$ matrix, where N is the number of nodes in a graph. Each element in $\mathbf{W}$ represents the existence of a link. An input vector $\mathbf{y}$ contains class labels for each node. In our case, $\mathbf{y}$ expresses the observation of an entity in the current utterance. In other words, y=1 expresses that an entity is observed in an utterance, and y=0 expresses unobserved entities in an utterance. $\mathbf{f}$ is a vector of the predicted class label of each node. The objective function of label propagation to be minimized is defined as,

$$J(f) = \sum_{i=1}^{n}(y_i - f_i)^2 + \lambda \sum_{i<j} w_{i,j}(f_i - f_j)^2. \tag{1}$$

The first term in Equation (1) approximates predicted values to get close to the input values of the same node. The second term approximates closer values for predicted values of neighboring nodes. λ is a constant value for keeping balance between the first and second terms.

The formula deformation of Equation (1) with the Laplacian matrix is,

$$J(f) = \|\mathbf{y} - \mathbf{f}\|_{\mathbf{2}}^{\mathbf{2}} + \lambda \mathbf{f}^{\mathbf{T}} \mathbf{L} \mathbf{f}. \tag{2}$$

$\mathbf{L} \equiv \mathbf{D} - \mathbf{W}$ is a Laplacian matrix, and $\mathbf{D}$ is the summation of each row into diagonal components. This minimization problem is solved with,

$$(\mathbf{I} + \lambda \mathbf{L})\mathbf{f} = \mathbf{y}, \tag{3}$$

as defined in [23].

We implemented Equation (3), where observed entities (=nodes) in utterances are vectorized as $\mathbf{y}$, and $\mathbf{f}$ is a vector of predicted values of relaxed class nodes inferred on a subgraph. Then, we calculate $\mathbf{f}$ by,

$$\mathbf{f} = \mathbf{y}(\mathbf{I} + \lambda \mathbf{L})^{-\mathbf{1}}. \tag{4}$$

An example of label propagation executed on a created subgraph is shown in Figure 3. We assume that *"Singapore"* and *"Malaysia"* are observed. The input values of these nodes are 1's, and the others are 0's, as shown on the left side of the arrows on each node. The values on the right side are outputs of the label propagation executed on the subgraph. As a result, the "Country" node gets the highest value among the unobserved nodes because the relations with observed nodes are stronger than the other nodes. Concretely, the "Country" and "Asia" nodes have relations to the observed nodes; however, the "Country" node has a 1-hop away relation to the "Singapore" node through redirection with the "Island Nation" node; thus, "Country" has higher value than "Asia".

3.4. Discounts on Dialog History

We consider a *dialog history* at the input of label propagation because the dialog state gradually changes while the dialog continues. At the current observed values, the previous values of $\mathbf{y}$ are also added with a discount value γ, which is a value between $0 \leq d \leq 1$, during a sub-dialog segment. Once the discount value is factored on the previous $\mathbf{y}$, the current $\mathbf{y}$ is replaced with the addition of the factored values and current values. The process including the propagation is shown in **Algorithm 1**.

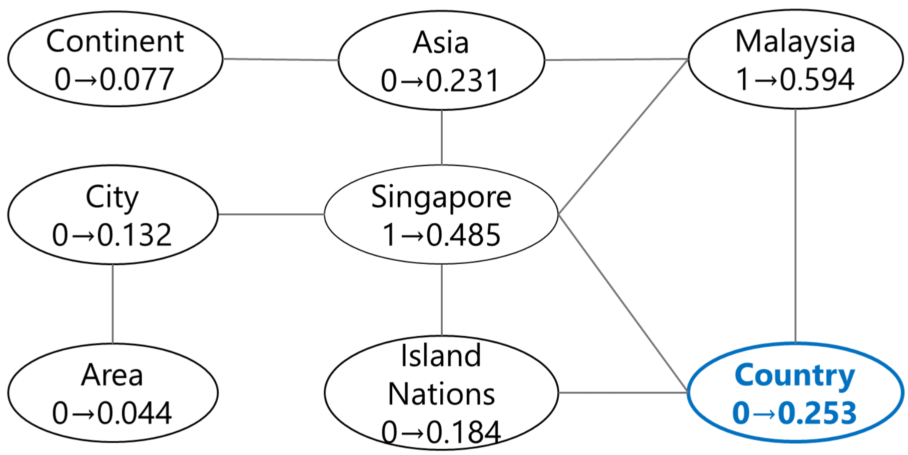

Figure 3: Example of label propagation on created subgraph

3.5. Dimension Reduction for Computational Efficiency

Dimension reduction based on principal contribution analysis (PCA) is applied to our feature vectors since they consume space and time complexity while tuning the tracker models. A subgraph created from the DSTC4 corpus contained more than 55,000 nodes, and the inferred feature vectors have the same dimension. Using this feature vector as is on neural networks causes there to be numerous parameters and a long calculation time. Thus, we applied PCA to the feature vector by keeping the cumulative contribution rate at 1. As a result, the size of the feature vectors was reduced to 2,500.

4. Neural-network-based Dialog State Trackers

In this section, we introduce several dialog state tracker models based on neural networks, which have been widely used in recent years. The first model is based on the fully connected neural network (FCNN) with our proposed feature vectors. The second model is based on the convolutional neural network (CNN), which achieved the highest score for DST among neural network models. We apply the ensemble methods based on the FCNN and CNN models in two different ways. One is the linear interpolation of outputs, and the other is the concatenation of the middle layers of neural networks.

Algorithm 1 Label Propagation with Discount Factor

Require: $\lambda > 0$, $0 \leq d \leq 1$, $i = index$ and $t = time$

```
if Initial Utterance in Sessions then
    for y_{i,t} in the word list do
        y_{i,t} = 1
    end for
else
    From second utterance do:
    for y_{i,t} do
        if y_{i,t} in the word list then
            y_{i,t} = 1 + γ y_{i,t-1}
        else
            y_{i,t} = γ y_{i,t-1}
        end if
    end for
end if
f = y(I + λL)^{-1}
return f
```

4.1. Fully Connected Neural Network with Inference on Knowledge Graph Feature Vectors

Our proposed feature vectors are used as inputs of the model of dialog state tracking. Recurrent or convolutional neural networks that can consider a sequence of words are widely used as dialogue state trackers; however, the meanings of sequences at each dimension of our inferred features vanish in the inference process. Thus, we use the FCNN as the classifier. This model consists of three-layers: an input layer, hidden layer, and output layer. The input layer size is 2,500 as the size of the compressed feature vectors. The hidden layer is also the same size as the input layer. The output layer size is 5,608 since all slot-value pairs are estimated at the same time. Sigmoid cross entropy is used as the loss function for multi-label classification.

4.2. Convolutional Neural-network-based Dialog State Tracker

A CNN-based dialog state tracker with word2vec is the state-of-the-art method for DSTC5 [12]. We exploit this model with a change in the output layer. This model takes the concatenation of the word vectors S to represent a sentence, and it obtains features for each sentence.

$$S = (word_1, word_2, ..., word_n). \tag{5}$$

S contains the concatenation of word vectors $word_i \in \mathbb{R}^k$. Each $word_i$ represents words in a sentence. i is the order of word occurrence in a sentence, and k is the dimension of the word vectors. Words are converted into word vectors by word2vec [13] by using a pretrained model of all Wikipedia articles. At the convolutional layer, the model contains a filter $w \in \mathbb{R}^{d \times k}$ for creating a feature $h_i \in \mathbb{R}^{n-d+1}$, where d is the filter height.

$$h_i = f(w * word_i : word_{i+d-1} + b). \tag{6}$$

b is a bias term, and f is a non-linear function. A feature map h is produced by using the $filter$ on each possible word in a sentence.

$$h = [h_1, h_2, ..., h_{n-d+1}]. \tag{7}$$

A max-over-time pooling is applied over the feature map to obtain the maximum value $\hat{h} = max\{h\}$ as the most important feature. The model may use multiple $filters$ to obtain multiple features by changing the filter height. After the max-over-time pooling, the features connect to the output layer through a fully connected layer. Sigmoid cross entropy is also used as the loss function of this model.

Our change from the CNN model, the best model for DSTC5, is the output layer. The CNN for DSTC5 is trained for each topic, and five models are produced as the number of topics. However, our model estimates all slot-value pairs at the same time. The data size of DSTC4 and DSTC5 is different; DSTC4 is smaller than DSTC5. Thus, we trained the model for all slot-value pairs at the same time to train an efficient model.

4.3. Ensemble of Inference-knowledge-feature-based Tracker and Convolutional Neural-network-based Trackers

We propose two ensemble models of the fully connected neural network with the proposed feature vectors and the convolutional neural network since these models take different features, which may cause estimation results to differ.

One ensemble model (*Ensemble-1*) combines FCNN and CNN outputs by linear interpolation. Additional weights (w_{fcnn} and w_{cnn}) are factored on both outputs (y_{fcnn} and y_{cnn}) because the trained FCNN and CNN have weights in different ranges.

$$\mathbf{y}_{ensemble_1} = \mathbf{y}_{fcnn} \times w_{fcnn} + \mathbf{y}_{cnn} \times w_{cnn} \tag{8}$$

w_{fcnn} and w_{cnn} satisfy $0 \leq w_{fcnn}, w_{cnn} \leq 1$, and $w_{fcnn} + w_{cnn} = 1$ to balance their different outputs. By using these weights, we are able to know which model has more factors for correct estimation.

The other model combines the hidden layers of both models, and the weights are simultaneously trained. We concatenate the hidden layer of FCNN and non-linear function on CNN feature $\hat{h}$ as,

$$\mathbf{h}_{ensemble_2} = \mathbf{h}_{fcnn} \otimes ReLU(\hat{c}_{cnn} * w + b). \tag{9}$$

This model ideally updates the weights between these features and the output layer at the training step. Intuitively, the model considers both features to estimates the slot-value pairs. The outputs of the model produced with a sigmoid function are,

$$\mathbf{y}_{ensemble_2} = \sigma(\mathbf{h}_{ensemble_2} * w + b). \tag{10}$$

4.4. Other Details on Neural Network Models

All introduced models use some common techniques for the convenience of model implementation. The output layer is a sigmoid function, so their loss function is sigmoid cross entropy. This is because our models estimate multiple slot-value pairs at the same time for multi-label classification. We utilize the Adam optimizer. Weight decay is used for all of the models. Dropout and batch normalization are used for each layer of all of the models.

5. Experiments

We conducted experiments on our proposed models with the DSTC4 dataset. Two different experimental settings were performed for each purpose. The first experiment was conducted to find the contribution of our proposed feature vectors on dialog state tracking. The second experiment was conducted to find the best model from the introduced models, including ensemble models with the state-of-the-art CNN model.

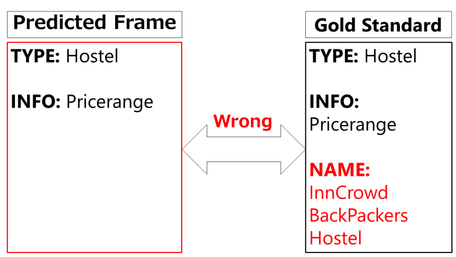

Figure 4: Calculating *Accuracy*

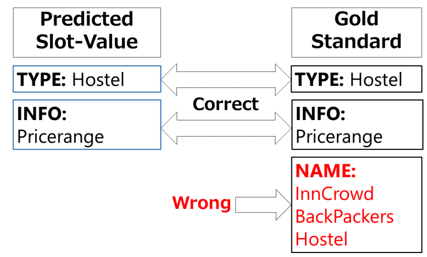

Figure 5: Calculating *F-measure*

5.1. Evaluation metrics

DSTC4 prepared two metrics and two schedules for calculating scores: *Accuracy* and *F-measure*. *Accuracy* calculates perfect matches of state frames. In other words, it recognizes that the state frame is incorrect when there are any extra or missing slot-value pairs. *F-measure* is the harmonic mean of *Precision* and *Recall*, which are calculated for each slot-value pair. Score matching examples of *Accuracy* and *F-measure* are illustrated in Figures 4 and 5. There are two different schedules prepared for calculating these metrics. One is called Schedule1, which calculates these metrics at each utterance, and the other is called Schedule2, which calculates these metrics only at the end of sub-dialog segments.

5.2. Experiment Based on Inference on Knowledge Graph

The first experiment was conducted to show the performance of our proposed method on dialog state tracking. The experiment consists of 2 steps: determining the best hyper-parameter combination and comparing our proposed feature with other features.

We explored the best hyper-parameter setting for the proposed method with the development data-set. To find the best hyper-parameter combinations, all combinations of Table 2 were examined for the experiment. The proposed method requires three hyper-parameters, which are λ in label propagation, γ as the weight of *dialog history*, and τ as the threshold of the output layer. λ is a parameter in label propagation that balances the first and second terms of Eqn. 1. γ is a discount rate of the input of label propagation. 0 means that there is no consideration of *dialog history*, and 1 means that a whole *dialog history* is considered. τ is a threshold at the output of the neural network model. It is set in strides of 0.1 from 0.1 to 0.9.

Table 2: List of Parameters for Proposed Method

λ	γ	τ
0.5	0	0.1
1	0.125	0.2
1.5	0.25	0.3
2	0.5	0.4
3	0.7	0.5
8	0.8	0.6
	0.9	0.7
	1	0.8
		0.9

Table 3: Top-5 *Accuracy* of FCNN-AKFV for Schedule1

λ	γ	τ	*Accuracy*
0.5	1.0	0.3	0.0490
0.5	1.0	0.2	0.0481
0.5	1.0	0.4	0.0456
1	1.0	0.3	0.0452
3	1.0	0.3	0.0427
Baseline			0.0374

Table 4: Top-5 *Accuracy* of FCNN-AKFV for Schedule2

λ	γ	τ	*Accuracy*
0.5	1.0	0.3	0.0559
0.5	1.0	0.2	0.0559
0.5	1.0	0.4	0.0549
0.5	1.0	0.5	0.0521
3	1.0	0.3	0.0502
Baseline			0.0488

Table 5: Top-5 *F-measure* of FCNN-AKFV for Schedule1

λ	γ	τ	*F-measure*
0.5	1.0	0.2	0.3444
3	1.0	0.2	0.3397
1	1.0	0.2	0.3391
8	1.0	0.2	0.3381
2	1.0	0.2	0.3371
Baseline			0.2506

Table 6: Top-5 *F-measure* of FCNN-AKFV for Schedule2

λ	γ	τ	*F-measure*
1	1.0	0.2	0.3759
3	1.0	0.2	0.3763
8	1.0	0.2	0.3754
0.5	1.0	0.2	0.3750
2	1.0	0.2	0.3727
Baseline			0.3014

Tables 3-6 shows the top-5 results from all parameter combinations. "Baseline" is the baseline system of DSTC4, which is implemented by *fuzzy string matching* by using only observable information in utterances. Specifically, Tables 3 and 4 show *Accuracies*, and Tables 5 and 6 show *F-measures* for

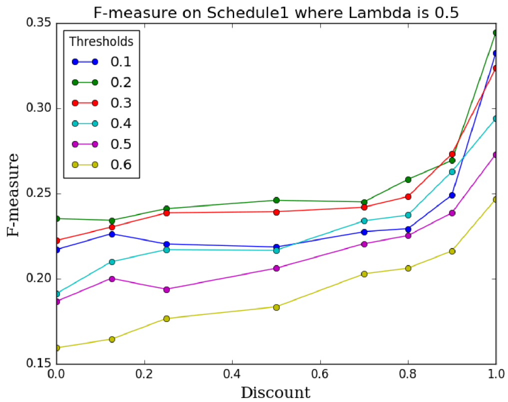

Figure 6: Changes on *F-measure* with discount rate on schedule1

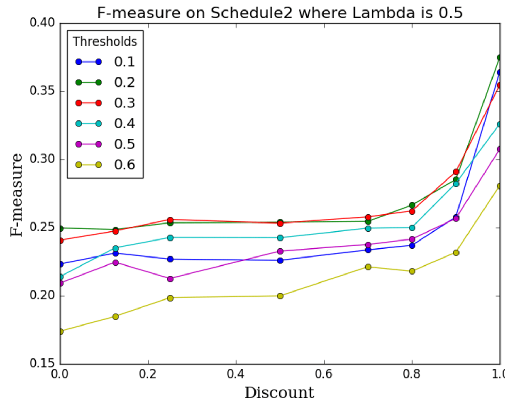

Figure 7: Changes on *F-measure* with discount rate on schedule2

each schedule. Our proposed method outperformed the baseline method for all comparisons. In accordance with the results, we set parameters: discount factor γ=1, weight λ=0.5, and threshold τ=0.2. According to the result tables, discount factor γ=1 achieves the highest results, and thus we could conclude that all histories without discounting contributed to the better results. Figure 6 and 7 respectively show changes of *F-measures* on schedule1 and schedule2 according to the changes of discount factor γ, and each curve represents the relation between the F-measure and the threshold. According to the figures, the impact of considering all histories are obvious since the increasing rate is over 0.1 from γ=0.0 to γ=1.0 on *F-measures* of both schedules.

To investigate the effectiveness of our proposed method, we compared the results of the associative knowledge graph feature vector (AKFV), conventional bag-of-words feature vector (BoW), and embedding feature vector based on word2vec (W2V). BoW is the most basic and simplest feature vector for natural language processing tasks. This feature only has simple information on sentences that represents the occurrences of words. W2V provides a fixed-length continuous feature vector. The training data used for the W2V model is external data from Wikipedia that we utilize as external data for our proposed method. It is linearly interpolated for all words appearing in a sentence. After the interpolation, it is inputted with BoW.

We compared three feature vectors by using the FCNN based dialog state

Table 7: Scores for Schedule1

	BoW	**W2V**	**AKFV**
Accuracy	0.004	0.010	0.039
Precision	0.010	0.285	0.445
Recall	0.044	0.066	0.275
F-measure	0.016	0.107	0.340

Table 8: Scores for Schedule2

	BoW	**W2V**	**AKFV**
Accuracy	0.007	0.013	0.050
Precision	0.009	0.2563	0.4596
Recall	0.534	0.074	0.319
F-measure	0.016	0.114	0.376

tracker. All scores of the vectors for on Schedule1 and Schedule2 are shown in Tables 7 and 8. Our proposed model, which is FCNN-AKFV, outperformed BoW and BoW with W2V, as shown in the tables. FCNN-AKFV was 36% higher than BoW and 26% higher than W2V scores for *F-measure*. In accordance with this score comparison, we investigated to find whether associative knowledge from observed words gains more features of sentences and more meaningful features for dialog state tracking.

5.3. Comparison with Other Neural-network-based Models

In this section, we discuss an experiment conducted to compare the neural-network-based models. We compared FCNN-AKFV, CNN, and two different ensemble models (*Ensemble-1* and *Ensemble-2*). We also compared the score results of the neural network-based approaches reported in DSTC4, which were the RNN [9] and CNN [11] trackers. Due to the score of the CNN tracker, it is the-state-of-the-art neural network model for DST; however, a W2V model for the CNN [11] tracker is trained by using Tripadvisor data, which cannot be currently used for research. Thus, we also show the score of the CNN-model with our implementation, which uses a W2V model trained from Wikipedia.

All of the models' hyper-parameters were tuned to the development set, and we applied top-4 models to the test set. The models had common hyper-parameters and their own hyper-parameters. The common hyper parameters were *dropout ratio* (**DR**), *threshold* (τ), *weight decay* (**WD**), α in Adam optimizer, and *batch size* (**BS**). **DR** and τ were set between 0.1 to 0.5 in increments of 0.1 for all models. **WD** was also set to be 0.000001. α in Adam optimizer and **BS** were uniquely set for all models. FCNN-AKFV required only these common hyper-parameters. α in Adam optimizer was 0.000025, and **BS** for FCNN was set between 20 to 110 in increment of 10. CNN required extra hyper-parameters, which were *filter hight* (**FH**) and *output channel* (**OC**). **FH** was 1, 2, or both, and **OC** was between 500 to 1,500 in increment of 500. α in Adam optimizer was 0.000001, and **BS** was set between 50 to 100 in increment of 25.

The ensemble models were implemented in different ways. *Ensemble-1* combined outputs of FCNN-AKFV and CNN, and all of the trained models were utilized weights. The weights were w_{fcnn} and w_{cnn}, which respectively factored on FCNN-AKFV and CNN. *Ensemble-2* simultaneously trained FCNN and CNN by concatenating hidden layers. We reduced the **FH** setting to be only 1 since all of the **FHs**, which were set to 2, had low scores from the overview of the CNN results on the development set. All of the other hyper-parameters were set with the same setting as the CNN, except **BS**. **BS** was set to be 25, 50, and 100.

Tables 9 and 10 show the scores for Schedule1 and Schedule2 for the test set. The *Ensemble-1* model outperformed all implemented models, RNN [9] and CNN [11]. The best sores of the implemented models are shown in the tables and were chosen by looking at *F-measure* for Schedule2.

The *Ensemble-1* model outperformed the other scores of the *F-measure* for Schedule1. It was 2.4% higher than the state-of-the-art CNN model [11] and 8.4% higher than the score of RNN [9]. *Ensemble-1* maintained *Precision* from the FCNN-AKFV model, and the score was 8.0% higher than the CNN model [11]. According to these score differences, the FCNN-AKFV and CNN models predicted different slot-value pairs from each feature vector. In other words, each feature vector contained different information for dialog state tracking. For Schedule2, *Ensemble1* had a similar tendency compared with the other results. Combining results estimated from different features caused a higher score for *Recall*.

FCNN-AKFV and *Ensemble-1* achieved significantly higher *Accuracy* than CNN [11] for Schedule1 ($p < 0.01$). *Ensemble-2* also achieved significantly higher *Accuracy* than CNN [11] for Schedule 1 ($p < 0.05$). The results for Schedule2 were not significant; however, the proposed models achieved com-

Table 9: Results of 4 models for Schedule1

	FCNN-AKFV	**CNN**	**Ensemble-1**
Accuracy	**0.046**	0.033	0.045
Precision	0.492	0.415	**0.495**
Recall	0.259	0.254	0.283
F-measure	0.339	0.315	**0.360**
	Ensemble-2	**CNN [11]**	**RNN [9]**
Accuracy	0.044	0.037	0.026
Precision	0.422	0.418	0.364
Recall	**0.346**	0.280	0.222
F-measure	0.338	0.336	0.276

Table 10: Results of 4 models for Schedule2

	FCNN-AKFV	**CNN**	**Ensemble-1**
Accuracy	0.050	0.041	0.056
Precision	0.508	0.437	**0.516**
Recall	0.306	0.298	0.334
F-measure	0.382	0.354	**0.405**
	Ensemble-2	**CNN [11]**	**RNN [9]**
Accuracy	0.044	**0.058**	0.042
Precision	0.422	0.438	0.373
Recall	**0.348**	0.343	0.292
F-measure	0.380	0.385	0.328

parable scores to CNN [11]. *Ensemble-1* outperformed the CNN model, which used resources comparable to the proposed FCNN-AKFV and ensemble models. It also outperformed RNN [9] for all metrics defined in DSTC4.

6. Analysis between Proposed Features and Results

In this section, we conducted two analysis: case analysis and correlation analysis. In the case analysis, we compared the state frame results of both experiments: section 5.2 and section 5.3. In the correlation analysis, correlation coefficients were provided by using all of the combinations between input features and slot-value pairs.

Table 11: Frame States of Baseline, Proposed Method, and Gold Standard

Transcription	**Baseline**	**FCNN − AKFV**	**GoldStandard**
Uh National Museum, you may even get free entry because it's a- if it's a public holiday.		'INFO': ['Fee']	'PLACE': ['National Museum of Singapore'], 'INFO': ['Fee']

6.1. Case Analysis

Table 11 and Table 12 respectively show examples of a state frame of section 5.2 and 5.3 by comparing with the gold standard. In Table 11, we compared the proposed method and the "Baseline" (*fuzzystring matching*) since it only estimates the slot-value pairs from observed information in utterances. We used the baseline system for the comparison instead of "Bag-of-Words" or "Word2Vec" because the score was higher than these method.

Table 11 shows the slot-values pairs of "Baseline" and FCNN-AKFV. FCNN-AKFV estimated a correct slot-value pair which was not estimated by "Baseline". The value was not observed in the utterance as a word. Concretely, the proposed method predicts the value **'Fee'** for the slot **'INFO'**. The word 'Fee' is not observed in the utterance; however, the proposed tracker successfully predicted the slot-value pair by using the proposed features, which is probably inferred from 'free entry' in the user utterance.

Table 12 shows examples that our ensemble models correctly estimated the state frame. We determined that there was a synergistic effect with the ensemble models. The state frame of ensemble models is exactly matched with gold standard for the utterance even though based models does not. Both single models only predict a correct slot-value pair; therefore, the synergy of these models enable to capture additional states: **'ACTIVITY'** and **'INFO'**.

6.2. Correlation Analysis

Correlation analysis provides the correlation coefficients to clarify effectiveness of our proposed feature vector. We provided correlation analysis of the typical case we discussed in Table 11. The correlation coefficients are calculated between AKFV and predicted results of FCNN-AKFV.

We firstly analyzed the general case of 'INFO':['Fee'] in Table 13. The table shows top 15 correlation coefficients to visualize the affect from the proposed features to the estimation results. The number at the end of each

Table 12: State Frames of neural network based models

Transcription	**Gold Standard**	**FCNN-AKFV**	**CNN**
also for certain rides in the Universal Studio, there's a height limit.	**PLACE**: 'Universal Studios Singapore' **ACTIVITY**: 'Amusement ride' **INFO**: 'Restriction'	**PLACE**: 'Universal Studios Singapore'	**PLACE**: 'Universal Studios Singapore'
	Ensemble-1	**Ensemble-2**	
	PLACE: 'Universal Studios Singapore' **ACTIVITY**: 'Amusement ride' **INFO**: 'Restriction'	**PLACE**: 'Universal Studios Singapore' **ACTIVITY**: 'Amusement ride' **INFO**: 'Restriction'	

word is the unique ID for each entities in Wikidata. The names start with 'Q' is the entity-id of Wikidata, but they contain some entities that have the same word surface in Wikidata. The entities of 'fee0'-'fee4' have higher correlation coefficients (0.481). It is not shown in the table, but similar entity 'cost' also has high correlation coefficients (0.276).

Table 14 shows the top 15 feature weights and related correlation coefficients between the AKFV and the FCNN-AKFV. We eliminated the input entities to show the easy overview of the related nodes affects. The most of the entities on the table were unobservable in the training set, and some features have high correlation coefficients to 'INFO':['Fee']. *Q6012465* has the highest weight and high correlation coefficient, this entity came from *neighbor nodes* of *free*. Q9099391, Q12221315, Q3960697, Q3053171 and Q6655391 are also related to *free* as *neighbor nodes*, which have high correlation coefficients. These results indicate that a lot of nodes, related to the 'free', contributed to estimate 'INFO':['Fee'].

7. Conclusion

In this paper, we proposed feature vector creation with associative knowledge through inference on a knowledge graph. We conducted experiments to show the effectiveness of the proposed feature vectors on a neural-network-based DST. The case analysis showed that the vectors were an effective approach for DST. We also proposed ensemble models based on FCNN-AKFV and the state-of-the-art CNN tracker. An ensemble model outperformed other neural network based DSTs. The correlation analysis indicated that the neighboring nodes inferred by the proposed method contributed to improve the result of trackers. As future work, we will analyze the creation and aspects of the graph for more effective graph creation. We will also approaches other inference methods acquiring associative knowledge. It is also considerable to jointly optimize the feature creation and the dialog state tracking. The proposed inference method to create associative feature will be helped on for other tasks of spoken language understanding, thus we plan to apply our method for a variety of tasks as future works.

8. Acknowledgement

Part of this research was supported by JST PRESTO (JPMJPR165B).

Table 14: Top 15 correlation coefficients of the case table 11

'INFO':['Fee']	**feat.**	**correl.**
Q11972 canton of Aargau	0.432	-0.007
Q6012465 *In the Meantime, In Between Time*	0.350	0.239
Q5215183 Dance Party in the Balkans	0.350	0.010
b0	0.346	-0.043
Q9099391 *Not Labeled*	0.346	0.239
Q16334295 group of humans	0.346	0.033
Q16421734 Maj	0.307	-0.035
Q12221315 *Not Labeled*	0.233	0.236
Q18553401 Soul Eater	0.231	0.239
Q20880814 The Groggers	0.231	-0.046
Q3960697 *Silver Rain*	0.231	0.239
Q914012 Planetshakers	0.231	0.239
Q3053171 *Emotional Playground*	0.231	0.233
Q507942 CTI Records	0.231	0.239
Q6655391 *Live Over Europe!*	0.231	0.239

Table 13: Top 15 correlation coefficients of 'INFO':['Fee']

	'INFO':['Fee']
entrance6	0.548
entrance7	0.546
entrance1	0.544
Q653475 (X display manager)	0.544
entrance2	0.544
entrance0	0.544
entrance5	0.543
entrance4	0.542
Q739937 (Declan Quinn)	0.516
Q1968839 (Paul Weitz)	0.516
Q6170802 (Jean Hanff Korelitz)	0.516
Q511731 (Imagine Entertainment)	0.516
admission0	0.516
much0	0.484

9. References

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

[2] B. Thomson, S. Young, Bayesian update of dialogue state: A pomdp framework for spoken dialogue systems, Computer Speech & Language 24 (2010) 562–588.

[3] J. Williams, A. Raux, M. Henderson, The dialog state tracking challenge series: A review, Dialogue & Discourse 7 (2016) 4–33.

[4] J. Williams, A. Raux, D. Ramachandran, A. Black, The dialog state tracking challenge, in: Proceedings of the Special Interest Group on Discource and Dialogue 2013, pp. 404–413.

[5] M. Henderson, B. Thomson, J. Williams, The second dialog state tracking challenge, in: Proceedings of the Special Interest Group on Discource and Dialogue 2014, pp. 263–272.

[6] M. Henderson, B. Thomson, J. Williams, The third dialog state tracking challenge, in: Proceedings of the Spoken Language Technology 2014, pp. 324–329.

[7] M. Henderson, Machine learning for dialog state tracking: A review, in: Proceedings of Machine Learning in Spoken Language Processing 2015.

[8] M. Henderson, B. Thomson, S. Young, Word-based dialog state tracking with recurrent neural networks, in: Proceedings of the Special Interest Group on Discource and Dialogue 2014, pp. 292–299.

[9] K. Yoshino, T. Hiraoka, G. Neubig, S. Nakamura, Dialog state tracking using long short term memory neural networks, in: Proceedings of International Workshop on Spoken Dialog Systems 2016.

[10] T. Hori, H. Wang, C. Hori, S. Watanabe, B. Harsham, J. Le Roux, J. R. Hershey, Y. Koji, Y. Jing, Z. Zhu, et al., Dialog state tracking with attention-based sequence-to-sequence learning, in: Proceedings of Spoken Language Technology Workshop 2016, pp. 552–558.

[11] H. Shi, T. Ushio, M. Endo, K. Yamagami, N. Horii, Convolutional neural networks for multi-topic dialog state tracking, in: Proceedings of International Workshop on Spoken Dialog Systems 2016.

[12] H. Shi, T. Ushino, M. Endo, K. Yamagami, N. Horii, A multichannel convolutional neural network for cross-language dialog state tracking, in: Proceedings of the Spoken Language Technology 2016, pp. 559–564.

[13] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, Distributed represenatational of words and phrases and their compositionality, in: Proceedings of Advances in Nueral Information Processing System 2013, pp. 3111–3119.

[14] J. Pennington, R. Socher, C. Manning, Glove: Global vectors for word representation, in: Proceedings of Empirical Methods in Natural Language Processing 2014, pp. 1532–1543.

[15] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, J. Taylor, Freebase: A collaboratively created graph database for structuring human knowledge, in: Proceedings of International Conference on Management of Data 2008, pp. 1247–1250.

[16] D. Vrandečić, M. Krötzsch, Wikidata: a free collaborative knowledgebase, Communications 57 (2014) 78–85.

[17] Y. Ma, P. Crook, R. Sarikayu, E. Fosler-Lussier, Knoewledge graph inference for spoken dialog system, in: Proceedings of International Conference on Acoustics, Speech and Signal Processing 2015, pp. 5346–5305.

[18] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: Advances in Neural Information Processing Systems 2016, pp. 3844–3852.

[19] Y.-N. Chen, D. Hakkani-Tür, G. Tür, J. Gao, L. Deng, End-to-end memory networks with knowledge carryover for multi-turn spoken language understanding., in: Proceedings of International Conference on Acoustics, Speech and Signal Processing 2016, pp. 3245–3249.

[20] H. He, A. Balakrishnan, M. Eric, P. Liang, Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings, in: Proceedings of Association for Computational Linguistics 2017.

[21] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of Empirical Methods on Natural Language Processing 2014, pp. 1746–1751.

[22] Y. Zhang, B. Wallace, A sensitive analysis of (and practitioners' guide to) convolutional neural networks for sentence classification, in: Proceedings of International Joint Conference on Natural Language Processing 2017, pp. 253–263.

[23] T. Kato, H. Kashima, M. Sugiyama, Robust label propagation on multiple networks, IEEE Transactions on Neural Networks 20 (2009) 35–44.