# Neural Belief Tracker: Data-Driven Dialogue State Tracking

**Nikola Mrkšić**[1], **Diarmuid Ó Séaghdha**[2]
**Tsung-Hsien Wen**[1], **Blaise Thomson**[2], **Steve Young**[1]
[1] University of Cambridge
[2] Apple Inc.
{nm480, thw28, sjy}@cam.ac.uk
{doseaghdha, blaisethom}@apple.com

## Abstract

One of the core components of modern spoken dialogue systems is the *belief tracker*, which estimates the user's goal at every step of the dialogue. However, most current approaches have difficulty scaling to larger, more complex dialogue domains. This is due to their dependency on either: **a)** Spoken Language Understanding models that require large amounts of annotated training data; or **b)** hand-crafted lexicons for capturing some of the linguistic variation in users' language. We propose a novel Neural Belief Tracking (NBT) framework which overcomes these problems by building on recent advances in representation learning. NBT models reason over pre-trained word vectors, learning to compose them into distributed representations of user utterances and dialogue context. Our evaluation on two datasets shows that this approach surpasses past limitations, matching the performance of state-of-the-art models which rely on hand-crafted semantic lexicons and outperforming them when such lexicons are not provided.

## 1 Introduction

Spoken dialogue systems (SDS) allow users to interact with computer applications through conversation. Task-based systems help users achieve goals such as finding restaurants or booking flights. The *dialogue state tracking* (DST) component of an SDS serves to interpret user input and update the *belief state*, which is the system's internal representation of the state of the conversation (Young et al., 2010). This is a probability distribution over dialogue states used by the downstream *dialogue manager* to decide which action the system should perform next (Su et al., 2016a,b); the system action is then verbalised by the natural language generator (Wen et al., 2015a,b; Dušek and Jurčíček, 2015).

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

Figure 1: Annotated dialogue states in a sample dialogue. Underlined words show rephrasings which are typically handled using semantic dictionaries.

The Dialogue State Tracking Challenge (DSTC) series of shared tasks has provided a common evaluation framework accompanied by labelled datasets (Williams et al., 2016). In this framework, the dialogue system is supported by a *domain ontology* which describes the range of user intents the system can process. The ontology defines a collection of *slots* and the *values* that each slot can take. The system must track the search constraints expressed by users (*goals* or *informable* slots) and questions the users ask about search results (*requests*), taking into account each user utterance (input via a speech recogniser) and the dialogue context (e.g., what the system just said). The example in Figure 1 shows the true state after each user utterance in a three-turn conversation. As can be seen in this example, DST models depend on identifying mentions of ontology items in user utterances. This becomes a non-trivial task when confronted with lexical variation, the dynamics of context and noisy automated speech recognition (ASR) output.

**FOOD=CHEAP:** [affordable, budget, low-cost, low-priced, inexpensive, cheaper, economic, ...]

**RATING=HIGH:** [best, high-rated, highly rated, top-rated, cool, chic, popular, trendy, ...]

**AREA=CENTRE:** [center, downtown, central, city centre, midtown, town centre, ...]

Figure 2: An example semantic dictionary with rephrasings for three ontology values in a *restaurant search* domain.

Traditional statistical approaches use separate Spoken Language Understanding (SLU) modules to address lexical variability within a single dialogue turn. However, training such models requires substantial amounts of domain-specific annotation. Alternatively, turn-level SLU and cross-turn DST can be coalesced into a single model to achieve superior belief tracking performance, as shown by Henderson et al. (2014d). Such coupled models typically rely on manually constructed semantic dictionaries to identify alternative mentions of ontology items that vary lexically or morphologically. Figure 2 gives an example of such a dictionary for three slot-value pairs. This approach, which we term *delexicalisation*, is clearly not scalable to larger, more complex dialogue domains. Importantly, the focus on English in DST research understates the considerable challenges that morphology poses to systems based on exact matching in morphologically richer languages such as Italian or German (see Vulić et al. (2017)).

In this paper, we present two new models, collectively called the Neural Belief Tracker (NBT) family. The proposed models couple SLU and DST, efficiently learning to handle variation without requiring any hand-crafted resources. To do that, NBT models move away from exact matching and instead reason entirely over pre-trained word vectors. The vectors making up the user utterance and preceding system output are first composed into intermediate representations. These representations are then used to decide which of the ontology-defined intents have been expressed by the user up to that point in the conversation.

To the best of our knowledge, NBT models are the first to successfully use pre-trained word vector spaces to improve the language understanding capability of belief tracking models. In evaluation on two datasets, we show that: **a)** NBT models match the performance of delexicalisation-based models which make use of hand-crafted semantic lexicons; and **b)** the NBT models significantly outperform those models when such resources are not available. Consequently, we believe this work proposes a framework better-suited to scaling belief tracking models for deployment in real-world dialogue systems operating over sophisticated application domains where the creation of such domain-specific lexicons would be infeasible.

## 2 Background

Models for probabilistic dialogue state tracking, or *belief tracking*, were introduced as components of spoken dialogue systems in order to better handle noisy speech recognition and other sources of uncertainty in understanding a user's goals (Bohus and Rudnicky, 2006; Williams and Young, 2007; Young et al., 2010). Modern dialogue management policies can learn to use a tracker's distribution over intents to decide whether to execute an action or request clarification from the user. As mentioned above, the DSTC shared tasks have spurred research on this problem and established a standard evaluation paradigm (Williams et al., 2013; Henderson et al., 2014b,a). In this setting, the task is defined by an *ontology* that enumerates the goals a user can specify and the attributes of entities that the user can request information about. Many different belief tracking models have been proposed in the literature, from generative (Thomson and Young, 2010) and discriminative (Henderson et al., 2014d) statistical models to rule-based systems (Wang and Lemon, 2013). To motivate the work presented here, we categorise prior research according to their reliance (or otherwise) on a separate SLU module for interpreting user utterances:[1]

**Separate SLU** Traditional SDS pipelines use Spoken Language Understanding (SLU) decoders to detect slot-value pairs expressed in the Automatic Speech Recognition (ASR) output. The downstream DST model then combines this information with the past dialogue context to update the belief state (Thomson and Young, 2010; Wang and Lemon, 2013; Lee and Kim, 2016; Perez, 2016; Perez and Liu, 2017; Sun et al., 2016; Jang et al., 2016; Shi et al., 2016; Dernoncourt et al., 2016; Liu and Perez, 2017; Vodolán et al., 2017).

[1]The best-performing models in DSTC2 all used both raw ASR output and the output of (potentially more than one) SLU decoders (Williams, 2014; Williams et al., 2016). This does not mean that those models are immune to the drawbacks identified here for the two model categories; in fact, they share the drawbacks of both.

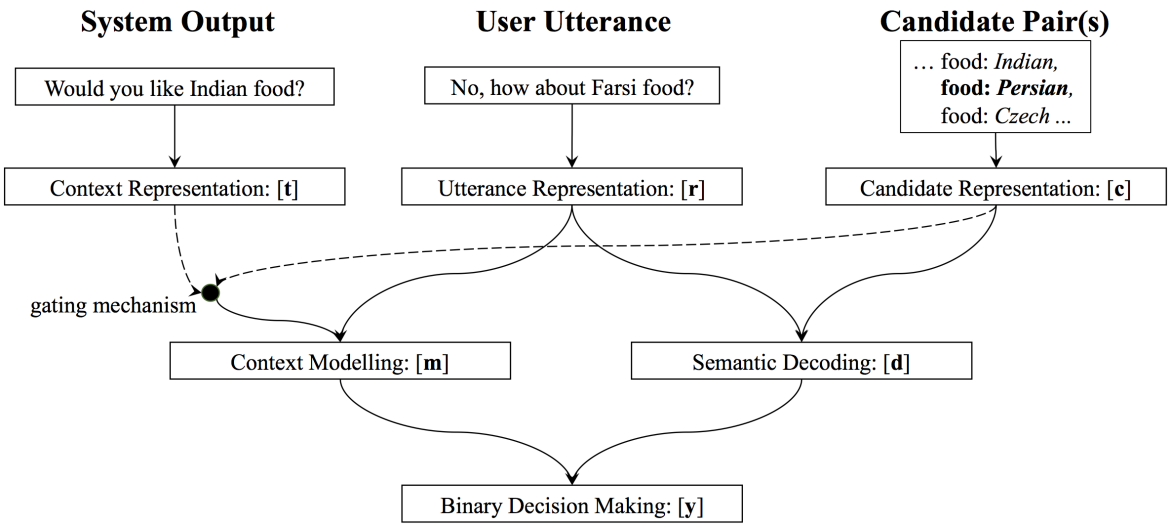


Figure 3: Architecture of the NBT Model. The implementation of the three representation learning subcomponents can be modified, as long as these produce adequate vector representations which the downstream model components can use to decide whether the current candidate slot-value pair was expressed in the user utterance (taking into account the preceding system act).

In the DSTC challenges, some systems used the output of template-based matching systems such as Phoenix (Wang, 1994). However, more robust and accurate statistical SLU systems are available. Many discriminative approaches to spoken dialogue SLU train independent binary models that decide whether each slot-value pair was expressed in the user utterance. Given enough data, these models can learn which lexical features are good indicators for a given value and can capture elements of paraphrasing (Mairesse et al., 2009). This line of work later shifted focus to robust handling of rich ASR output (Henderson et al., 2012; Tur et al., 2013). SLU has also been treated as a sequence labelling problem, where each word in an utterance is labelled according to its role in the user's intent; standard labelling models such as CRFs or Recurrent Neural Networks can then be used (Raymond and Ricardi, 2007; Yao et al., 2014; Celikyilmaz and Hakkani-Tur, 2015; Mesnil et al., 2015; Peng et al., 2015; Zhang and Wang, 2016; Liu and Lane, 2016b; Vu et al., 2016; Liu and Lane, 2016a, i.a.). Other approaches adopt a more complex modelling structure inspired by semantic parsing (Saleh et al., 2014; Vlachos and Clark, 2014). One drawback shared by these methods is their resource requirements, either because they need to learn independent parameters for each slot and value or because they need fine-grained manual annotation at the word level. This hinders scaling to larger, more realistic application domains.

**Joint SLU/DST** Research on belief tracking has found it advantageous to reason about SLU and DST jointly, taking ASR predictions as input and generating belief states as output (Henderson et al., 2014d; Sun et al., 2014; Zilka and Jurcicek, 2015; Mrkšić et al., 2015). In DSTC2, systems which used no external SLU module outperformed all systems that only used external SLU features. Joint models typically rely on a strategy known as *delexicalisation* whereby slots and values mentioned in the text are replaced with generic labels. Once the dataset is transformed in this manner, one can extract a collection of template-like $n$-gram features such as **[want *tagged-value* food]**. To perform belief tracking, the shared model iterates over all slot-value pairs, extracting delexicalised feature vectors and making a separate binary decision regarding each pair. Delexicalisation introduces a hidden dependency that is rarely discussed: how do we identify slot/value mentions in text? For toy domains, one can manually construct *semantic dictionaries* which list the potential rephrasings for all slot values. As shown by Mrkšić et al. (2016), the use of such dictionaries is essential for the performance of current delexicalisation-based models. Again though, this will not scale to the rich variety of user language or to general domains.

The primary motivation for the work presented in this paper is to overcome the limitations that affect previous belief tracking models. The NBT model efficiently learns from the avail-

able data by: **1)** leveraging semantic information from pre-trained word vectors to resolve lexical/morphological ambiguity; **2)** maximising the number of parameters shared across ontology values; and **3)** having the flexibility to learn domain-specific paraphrasings and other kinds of variation that make it infeasible to rely on exact matching and delexicalisation as a robust strategy.

## 3 Neural Belief Tracker

The Neural Belief Tracker (NBT) is a model designed to detect the slot-value pairs that make up the user's goal at a given turn during the flow of dialogue. Its input consists of the system dialogue acts preceding the user input, the user utterance itself, and a single candidate slot-value pair that it needs to make a decision about. For instance, the model might have to decide whether the goal FOOD=ITALIAN has been expressed in *'I'm looking for good pizza'*. To perform belief tracking, the NBT model *iterates* over all candidate slot-value pairs (defined by the ontology), and decides which ones have just been expressed by the user.

Figure 3 presents the flow of information in the model. The first layer in the NBT hierarchy performs representation learning given the three model inputs, producing vector representations for the user utterance ($\mathbf{r}$), the current candidate slot-value pair ($\mathbf{c}$) and the system dialogue acts ($\mathbf{t_q}, \mathbf{t_s}, \mathbf{t_v}$). Subsequently, the learned vector representations interact through the *context modelling* and *semantic decoding* submodules to obtain the intermediate *interaction summary* vectors $\mathbf{d_r}, \mathbf{d_c}$ and $\mathbf{d}$. These are used as input to the final *decision-making* module which decides whether the user expressed the intent represented by the candidate slot-value pair.

### 3.1 Representation Learning

For any given user utterance, system act(s) and candidate slot-value pair, the representation learning submodules produce vector representations which act as input for the downstream components of the model. All representation learning subcomponents make use of pre-trained collections of word vectors. As shown by Mrkšić et al. (2016), specialising word vectors to express *semantic similarity* rather than *relatedness* is essential for improving belief tracking performance. For this reason, we use the semantically-specialised Paragram-SL999 word vectors (Wieting et al., 2015) throughout this work. The NBT training procedure keeps these vectors fixed: that way, at test time, unseen words semantically related to familiar slot values (i.e. *inexpensive* to *cheap*) will be recognised purely by their position in the original vector space (see also Rocktäschel et al. (2016)). This means that the NBT model parameters can be shared across all values of the given slot, or even across all slots.

Let $u$ represent a user utterance consisting of $k_u$ words $u_1, u_2, \ldots, u_{k_u}$. Each word has an associated word vector $\mathbf{u}_1, \ldots, \mathbf{u}_{k_u}$. We propose two model variants which differ in the method used to produce vector representations of $u$: NBT-DNN and NBT-CNN. Both act over the constituent $n$-grams of the utterance. Let $\mathbf{v}_i^n$ be the concatenation of the $n$ word vectors starting at index $i$, so that:

$$\mathbf{v}_i^n = \mathbf{u}_i \oplus \ldots \oplus \mathbf{u}_{i+n-1} \tag{1}$$

where $\oplus$ denotes vector concatenation. The simpler of our two models, which we term NBT-DNN, is shown in Figure 4. This model computes cumulative $n$-gram representation vectors $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$, which are the $n$-gram 'summaries' of the unigrams, bigrams and trigrams in the user utterance:

$$\mathbf{r}_n = \sum_{i=1}^{k_u - n + 1} \mathbf{v}_i^n \tag{2}$$

Each of these vectors is then non-linearly mapped to intermediate representations of the same size:

$$\mathbf{r}'_n = \sigma(W_n^s \mathbf{r}_n + b_n^s) \tag{3}$$

where the weight matrices and bias terms map the cumulative $n$-grams to vectors of the same dimensionality and $\sigma$ denotes the sigmoid activation function. We maintain a separate set of parameters for each slot (indicated by superscript $s$). The three vectors are then summed to obtain a single representation for the user utterance:

$$\mathbf{r} = \mathbf{r}'_1 + \mathbf{r}'_2 + \mathbf{r}'_3 \tag{4}$$

The cumulative $n$-gram representations used by this model are just unweighted sums of all word vectors in the utterance. Ideally, the model should learn to recognise which parts of the utterance are more relevant for the subsequent classification task. For instance, it could learn to ignore verbs or stop words and pay more attention to adjectives and nouns which are more likely to express slot values.

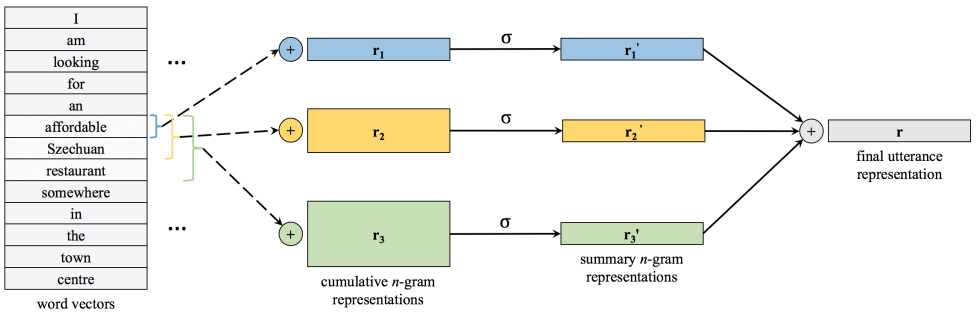


Figure 4: NBT-DNN MODEL. Word vectors of $n$-grams ($n = 1, 2, 3$) are summed to obtain *cumulative* $n$-grams, then passed through another hidden layer and summed to obtain the utterance representation $\mathbf{r}$.

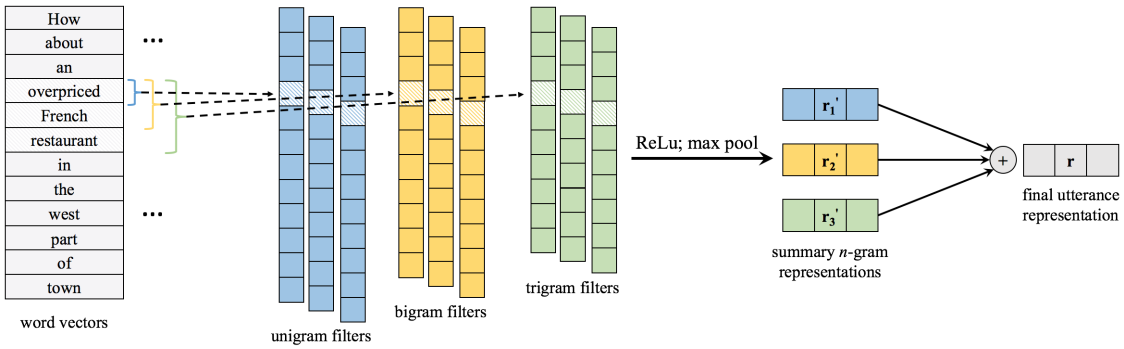


Figure 5: NBT-CNN Model. $L$ convolutional filters of window sizes $1, 2, 3$ are applied to word vectors of the given utterance ($L = 3$ in the diagram, but $L = 300$ in the system). The convolutions are followed by the ReLU activation function and max-pooling to produce summary $n$-gram representations. These are summed to obtain the utterance representation $\mathbf{r}$.

**NBT-CNN** Our second model draws inspiration from successful applications of Convolutional Neural Networks (CNNs) for language understanding (Collobert et al., 2011; Kalchbrenner et al., 2014; Kim, 2014). These models typically apply a number of convolutional filters to $n$-grams in the input sentence, followed by non-linear activation functions and max-pooling. Following this approach, the NBT-CNN model applies $L = 300$ different filters for $n$-gram lengths of $1, 2$ and $3$ (Figure 5). Let $F_n^s \in R^{L \times nD}$ denote the collection of filters for each value of $n$, where $D = 300$ is the word vector dimensionality. If $\mathbf{v}_i^n$ denotes the concatenation of $n$ word vectors starting at index $i$, let $\mathbf{m}_n = [\mathbf{v}_1^n; \mathbf{v}_2^n; \ldots; \mathbf{v}_{k_u-n+1}^n]$ be the list of $n$-grams that convolutional filters of length $n$ run over. The three intermediate representations are then given by:

$$R_n = F_n^s \, \mathbf{m}_n \tag{5}$$

Each column of the intermediate matrices $R_n$ is produced by a single convolutional filter of length $n$. We obtain summary $n$-gram representations by pushing these representations through a rectified linear unit (ReLU) activation function (Nair and Hinton, 2010) and max-pooling over time (i.e. columns of the matrix) to get a single feature for each of the $L$ filters applied to the utterance:

$$\mathbf{r}'_n = \texttt{maxpool}\left(\texttt{ReLU}\left(R_n + b_n^s\right)\right) \tag{6}$$

where $b_n^s$ is a bias term broadcast across all filters. Finally, the three summary $n$-gram representations are summed to obtain the final utterance representation vector $\mathbf{r}$ (as in Equation 4). The NBT-CNN model is (by design) better suited to longer utterances, as its convolutional filters interact directly with subsequences of the utterance, and not just their noisy summaries given by the NBT-DNN's cumulative $n$-grams.

### 3.2 Semantic Decoding

The NBT diagram in Figure 3 shows that the utterance representation $\mathbf{r}$ and the candidate slot-value pair representation $\mathbf{c}$ directly interact through the *semantic decoding* module. This component decides whether the user explicitly expressed an intent matching the current candidate pair

(i.e. without taking the dialogue context into account). Examples of such matches would be *'I want Thai food'* with `food=Thai` or more demanding ones such as *'a pricey restaurant'* with `price=expensive`. This is where the use of high-quality pre-trained word vectors comes into play: a delexicalisation-based model could deal with the former example but would be helpless in the latter case, unless a human expert had provided a semantic dictionary listing all potential rephrasings for each value in the domain ontology.

Let the vector space representations of a candidate pair's slot name and value be given by $\mathbf{c_s}$ and $\mathbf{c_v}$ (with vectors of multi-word slot names/values summed together). The NBT model learns to map this tuple into a single vector $\mathbf{c}$ of the same dimensionality as the utterance representation $\mathbf{r}$. These two representations are then forced to interact in order to learn a similarity metric which discriminates between interactions of utterances with slot-value pairs that they either do or do not express:

$$\mathbf{c} = \sigma\big(W_c^s(\mathbf{c_s} + \mathbf{c_v}) + b_c^s\big) \quad (7)$$

$$\mathbf{d} = \mathbf{r} \otimes \mathbf{c} \quad (8)$$

where $\otimes$ denotes *element-wise* vector multiplication. The dot product, which may seem like the more intuitive similarity metric, would reduce the rich set of features in $\mathbf{d}$ to a single scalar. The element-wise multiplication allows the downstream network to make better use of its parameters by learning non-linear interactions between sets of features in $\mathbf{r}$ and $\mathbf{c}$.[2]

### 3.3 Context Modelling

This 'decoder' does not yet suffice to extract intents from utterances in human-machine dialogue. To understand some queries, the belief tracker must be aware of *context*, i.e. the flow of dialogue leading up to the latest user utterance. While all previous system and user utterances are important, the most relevant one is the last system utterance, in which the dialogue system could have performed (among others) one of the following two *system acts*:

1. **System Request**: The system asks the user about the value of a specific slot $T_q$. If the system utterance is: *'what price range would you like?'* and the user answers with *any*, the model must infer the reference to *price range*, and not to other slots such as *area* or *food*.

2. **System Confirm:** The system asks the user to confirm whether a specific slot-value pair $(T_s, T_v)$ is part of their desired constraints. For example, if the user responds to *'how about Turkish food?'* with *'yes'*, the model must be aware of the system act in order to correctly update the belief state.

If we make the Markovian decision to only consider the last set of system acts, we can incorporate context modelling into the NBT. Let $\mathbf{t_q}$ and $(\mathbf{t_s}, \mathbf{t_v})$ be the word vectors of the arguments for the system request and confirm acts (zero vectors if none). The model computes the following measures of similarity between the system acts, candidate pair $(\mathbf{c_s}, \mathbf{c_v})$ and utterance representation $\mathbf{r}$:

$$\mathbf{m_r} = (\mathbf{c_s} \cdot \mathbf{t_q})\mathbf{r} \quad (9)$$

$$\mathbf{m_c} = (\mathbf{c_s} \cdot \mathbf{t_s})(\mathbf{c_v} \cdot \mathbf{t_v})\mathbf{r} \quad (10)$$

where $\cdot$ denotes dot product. The computed similarity terms act as gating mechanisms which only pass the utterance representation through if the system asked about the current candidate slot or slot-value pair. This type of interaction is particularly useful for the confirm system act: if the system asks the user to confirm, the user is likely not to mention any slot values, but to just respond affirmatively or negatively. This means that the model must consider the *three-way interaction* between the utterance, candidate slot-value pair and the slot value pair offered by the system. If (and only if) the latter two are the same should the model consider the affirmative or negative polarity of the user utterance when making the subsequent binary decision.

**Binary Decision Maker** The intermediate representations are passed through another hidden layer and then combined. If $\phi_{dim}(\mathbf{x}) = \sigma(W\mathbf{x} + b)$ is a layer which maps input vector $\mathbf{x}$ to a vector of size $dim$, the input to the final binary softmax (which represents the decision) is given by:

$$\mathbf{y} = \phi_2\big(\phi_{100}(\mathbf{d}) + \phi_{100}(\mathbf{m_r}) + \phi_{100}(\mathbf{m_c})\big)$$

## 4 Belief State Update Mechanism

In spoken dialogue systems, belief tracking models operate over the output of automatic speech recognition (ASR). Despite improvements to speech

[2] We also tried to concatenate $\mathbf{r}$ and $\mathbf{c}$ and pass that vector to the downstream decision-making neural network. However, this set-up led to very weak performance since our relatively small datasets did not suffice for the network to learn to model the interaction between the two feature vectors.

recognition, the need to make the most out of imperfect ASR will persist as dialogue systems are used in increasingly noisy environments.

In this work, we define a simple rule-based belief state update mechanism which can be applied to ASR $N$-best lists. For dialogue turn $t$, let $sys^{t-1}$ denote the preceding system output, and let $h^t$ denote the list of $N$ ASR hypotheses $h_i^t$ with posterior probabilities $p_i^t$. For any hypothesis $h_i^t$, slot $s$ and slot value $v \in V_s$, NBT models estimate $\mathbb{P}(s, v \mid h_i^t, sys^{t-1})$, which is the (turn-level) probability that $(s, v)$ was expressed in the given hypothesis. The predictions for $N$ such hypotheses are then combined as:

$$\mathbb{P}(s, v \mid h^t, sys^{t-1}) = \sum_{i=1}^{N} p_i^t \, \mathbb{P}\left(s, v \mid h_i^t, sys^t\right)$$

This turn-level belief state estimate is then combined with the (cumulative) belief state up to time $(t-1)$ to get the updated belief state estimate:

$$\begin{aligned}\mathbb{P}(s, v \mid h^{1:t}, sys^{1:t-1}) \; = \; & \lambda \, \mathbb{P}\left(s, v \mid h^t, sys^{t-1}\right) \\ & + (1-\lambda) \, \mathbb{P}\left(s, v \mid h^{1:t-1}, sys^{1:t-2}\right)\end{aligned}$$

where $\lambda$ is the coefficient which determines the relative weight of the turn-level and previous turns' belief state estimates.[3] For slot $s$, the set of its *detected values* at turn $t$ is then given by:

$$V_s^t = \{v \in V_s \mid \mathbb{P}\left(s, v \mid h^{1:t}, sys^{1:t-1}\right) \geq 0.5\}$$

For informable (i.e. goal-tracking) slots, the value in $V_s^t$ with the highest probability is chosen as the current goal (if $V_s^t \neq \{\emptyset\}$). For requests, all slots in $V_{req}^t$ are deemed to have been requested. As requestable slots serve to model single-turn user queries, they require no belief tracking across turns.

## 5 Experiments

### 5.1 Datasets

Two datasets were used for training and evaluation. Both consist of user conversations with task-oriented dialogue systems designed to help users find suitable restaurants around Cambridge, UK. The two corpora share the same domain ontology, which contains three *informable* (i.e. goal-tracking) slots: FOOD, AREA and PRICE. The users can specify values for these slots in order to find restaurants which best meet their criteria. Once the system suggests a restaurant, the users can ask about the values of up to eight *requestable* slots (PHONE NUMBER, ADDRESS, etc.). The two datasets are:

1. **DSTC2**: We use the transcriptions, ASR hypotheses and turn-level semantic labels provided for the Dialogue State Tracking Challenge 2 (Henderson et al., 2014a). The official transcriptions contain various spelling errors which we corrected manually; the cleaned version of the dataset is available at `mi.eng.cam.ac.uk/~nm480/dstc2-clean.zip`. The training data contains 2207 dialogues and the test set consists of 1117 dialogues. We train NBT models on transcriptions but report belief tracking performance on test set ASR hypotheses provided in the original challenge.

2. **WOZ 2.0**: Wen et al. (2017) performed a Wizard of Oz style experiment in which Amazon Mechanical Turk users assumed the role of the system or the user of a task-oriented dialogue system based on the DSTC2 ontology. Users typed instead of using speech, which means performance in the WOZ experiments is more indicative of the model's capacity for semantic understanding than its robustness to ASR errors. Whereas in the DSTC2 dialogues users would quickly adapt to the system's (lack of) language understanding capability, the WOZ experimental design gave them freedom to use more sophisticated language. We expanded the original WOZ dataset from Wen et al. (2017) using the same data collection procedure, yielding a total of 1200 dialogues. We divided these into 600 training, 200 validation and 400 test set dialogues. The WOZ 2.0 dataset is available at `mi.eng.cam.ac.uk/~nm480/woz_2.0.zip`.

**Training Examples** The two corpora are used to create training data for two separate experiments. For each dataset, we iterate over all train set utterances, generating one example for *each* of the slot-value pairs in the ontology. An example consists of a transcription, its context (i.e. list of preceding system acts) and a candidate slot-value pair. The binary label for each example indicates whether or not its utterance and context express the example's candidate pair. For instance, '*I would like Irish*

[3] This coefficient was tuned on the DSTC2 development set. The best performance was achieved with $\lambda = 0.55$.

*food*' would generate a positive example for candidate pair FOOD=IRISH, and a negative example for every other slot-value pair in the ontology.

**Evaluation** We focus on two key evaluation metrics introduced in (Henderson et al., 2014a):

1. **Goals** ('joint goal accuracy'): the proportion of dialogue turns where all the user's search goal constraints were correctly identified;

2. **Requests**: similarly, the proportion of dialogue turns where user's requests for information were identified correctly.

### 5.2 Models

We evaluate two NBT model variants: NBT-DNN and NBT-CNN. To train the models, we use the Adam optimizer (Kingma and Ba, 2015) with cross-entropy loss, backpropagating through all the NBT subcomponents while keeping the pre-trained word vectors fixed (in order to allow the model to deal with unseen words at test time). The model is trained separately for each slot. Due to the high class bias (most of the constructed examples are negative), we incorporate a fixed number of positive examples in each mini-batch.[4]

**Baseline Models** For each of the two datasets, we compare the NBT models to:

1. A baseline system that implements a well-known competitive delexicalisation-based model for that dataset. For DSTC2, the model is that of Henderson et al. (2014c; 2014d). This model is an $n$-gram based neural network model with recurrent connections between turns (but not inside utterances) which replaces occurrences of slot names and values with generic delexicalised features. For WOZ 2.0, we compare the NBT models to a more sophisticated belief tracking model presented in (Wen et al., 2017). This model uses an RNN for belief state updates and a CNN for turn-level feature extraction. Unlike NBT-CNN, their CNN operates not over vectors, but over delexicalised features akin to those used by Henderson et al. (2014c).

2. The same baseline model supplemented with a task-specific semantic dictionary (produced by the baseline system creators). The two dictionaries are available at `mi.eng.cam.ac.uk/~nm480/sem-dict.zip`. The DSTC2 dictionary contains only three rephrasings. Nonetheless, the use of these rephrasings translates to substantial gains in DST performance (see Sect. 6.1). We believe this result supports our claim that the vocabulary used by Mechanical Turkers in DSTC2 was constrained by the system's inability to cope with lexical variation and ASR noise. The WOZ dictionary includes 38 rephrasings, showing that the unconstrained language used by Mechanical Turkers in the Wizard-of-Oz setup requires more elaborate lexicons.

Both baseline models map exact matches of ontology-defined intents (and their lexicon-specified rephrasings) to one-hot delexicalised $n$-gram features. This means that pre-trained vectors cannot be incorporated directly into these models.

## 6 Results

### 6.1 Belief Tracking Performance

Table 1 shows the performance of NBT models trained and evaluated on DSTC2 and WOZ 2.0 datasets. The NBT models outperformed the baseline models in terms of both joint goal and request accuracies. For goals, the gains are *always* statistically significant (paired $t$-test, $p < 0.05$). Moreover, there was no statistically significant variation between the NBT and the lexicon-supplemented models, showing that the NBT can handle semantic relations which otherwise had to be explicitly encoded in semantic dictionaries.

While the NBT performs well across the board, we can compare its performance on the two datasets to understand its strengths. The improvement over the baseline is greater on WOZ 2.0, which corroborates our intuition that the NBT's ability to learn linguistic variation is vital for this dataset containing longer sentences, richer vocabulary and no ASR errors. By comparison, the language of the subjects in the DSTC2 dataset is less rich, and compensating for ASR errors is the main hurdle: given access to the DSTC2 test set transcriptions, the NBT models' goal accuracy rises to 0.96. This

[4] Model hyperparameters were tuned on the respective validation sets. For both datasets, the initial Adam learning rate was set to 0.001, and $\frac{1}{8}$th of positive examples were included in each mini-batch. The batch size did not affect performance: it was set to 256 in all experiments. Gradient clipping (to $[-2.0, 2.0]$) was used to handle exploding gradients. Dropout (Srivastava et al., 2014) was used for regularisation (with 50% dropout rate on all intermediate representations). Both NBT models were implemented in TensorFlow (Abadi et al., 2015).

| DST Model | DSTC2 | | WOZ 2.0 | |
|---|---|---|---|---|
| | Goals | Requests | Goals | Requests |
| **Delexicalisation-Based Model** | 69.1 | 95.7 | 70.8 | 87.1 |
| **Delexicalisation-Based Model + Semantic Dictionary** | 72.9* | 95.7 | 83.7* | 87.6 |
| **NEURAL BELIEF TRACKER: NBT-DNN** | 72.6* | 96.4 | **84.4*** | 91.2* |
| **NEURAL BELIEF TRACKER: NBT-CNN** | **73.4*** | **96.5** | 84.2* | **91.6*** |

Table 1: DSTC2 and WOZ 2.0 test set accuracies for: **a)** joint goals; and **b)** turn-level requests. The asterisk indicates statistically significant improvement over the baseline trackers (paired $t$-test; $p < 0.05$).

indicates that future work should focus on better ASR compensation if the model is to be deployed in environments with challenging acoustics.

### 6.2 The Importance of Word Vector Spaces

The NBT models use the semantic relations embedded in the pre-trained word vectors to handle semantic variation and produce high-quality intermediate representations. Table 2 shows the performance of NBT-CNN[5] models making use of three different word vector collections: **1)** 'random' word vectors initialised using the XAVIER initialisation (Glorot and Bengio, 2010); **2)** distributional GloVe vectors (Pennington et al., 2014), trained using co-occurrence information in large textual corpora; and **3)** *semantically specialised* Paragram-SL999 vectors (Wieting et al., 2015), which are obtained by injecting *semantic similarity constraints* from the Paraphrase Database (Ganitkevitch et al., 2013) into the distributional GloVe vectors in order to improve their semantic content.

The results in Table 2 show that the use of semantically specialised word vectors leads to considerable performance gains: Paragram-SL999 vectors (significantly) outperformed GloVe and XAVIER vectors for goal tracking on both datasets. The gains are particularly robust for noisy DSTC2 data, where both collections of pre-trained vectors consistently outperformed random initialisation. The gains are weaker for the noise-free WOZ 2.0 dataset, which seems to be large (and clean) enough for the NBT model to learn task-specific rephrasings and compensate for the lack of semantic content in the word vectors. For this dataset, GloVe vectors do not improve over the randomly initialised ones. We believe this happens because distributional models keep related, yet antonymous words close together (e.g. *north* and *south*, *expensive* and *inexpensive*), offsetting the useful semantic content embedded in this vector spaces.

[5]The NBT-DNN model showed the same trends. For brevity, Table 2 presents only the NBT-CNN figures.

| Word Vectors | DSTC2 | | WOZ 2.0 | |
|---|---|---|---|---|
| | Goals | Requests | Goals | Requests |
| **XAVIER (No Info.)** | 64.2 | 81.2 | 81.2 | 90.7 |
| **GloVe** | 69.0* | 96.4* | 80.1 | 91.4 |
| **Paragram-SL999** | **73.4*** | **96.5*** | **84.2*** | **91.6** |

Table 2: DSTC2 and WOZ 2.0 test set performance (*joint goals* and *requests*) of the NBT-CNN model making use of three different word vector collections. The asterisk indicates statistically significant improvement over the baseline XAVIER (random) word vectors (paired $t$-test; $p < 0.05$).

## 7 Conclusion

In this paper, we have proposed a novel neural belief tracking (NBT) framework designed to overcome current obstacles to deploying dialogue systems in real-world dialogue domains. The NBT models offer the known advantages of coupling Spoken Language Understanding and Dialogue State Tracking, without relying on hand-crafted semantic lexicons to achieve state-of-the-art performance. Our evaluation demonstrated these benefits: the NBT models match the performance of models which make use of such lexicons and vastly outperform them when these are not available. Finally, we have shown that the performance of NBT models improves with the semantic quality of the underlying word vectors. To the best of our knowledge, we are the first to move past intrinsic evaluation and show that *semantic specialisation* boosts performance in downstream tasks.

In future work, we intend to explore applications of the NBT for multi-domain dialogue systems, as well as in languages other than English that require handling of complex morphological variation.

## Acknowledgements

The authors would like to thank Ivan Vulić, Ulrich Paquet, the Cambridge Dialogue Systems Group and the anonymous ACL reviewers for their constructive feedback and helpful discussions.

## References


Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems.

Dan Bohus and Alex Rudnicky. 2006. A "k hypotheses + other" belief updating model. In *Proceedings of the AAAI Workshop on Statistical and Empirical Methods in Spoken Dialogue Systems*.

Asli Celikyilmaz and Dilek Hakkani-Tur. 2015. Convolutional Neural Network Based Semantic Tagging with Entity Embeddings. In *Proceedings of NIPS Workshop on Machine Learning for Spoken Language Understanding and Interaction*.

Ronan Collobert, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12:2493–2537.

Franck Dernoncourt, Ji Young Lee, Trung H. Bui, and Hung H. Bui. 2016. Robust dialog state tracking for large ontologies. In *Proceedings of IWSDS*.

Ondřej Dušek and Filip Jurčíček. 2015. Training a Natural Language Generator From Unaligned Data. In *Proceedings of ACL*.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of NAACL HLT*.

Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of AISTATS*.

Matthew Henderson, Milica Gašić, Blaise Thomson, Pirros Tsiakoulis, Kai Yu, and Steve Young. 2012. Discriminative Spoken Language Understanding Using Word Confusion Networks. In *Spoken Language Technology Workshop, 2012. IEEE*.

Matthew Henderson, Blaise Thomson, and Jason D. Wiliams. 2014a. The Second Dialog State Tracking Challenge. In *Proceedings of SIGDIAL*.

Matthew Henderson, Blaise Thomson, and Jason D. Wiliams. 2014b. The Third Dialog State Tracking Challenge. In *Proceedings of IEEE SLT*.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014c. Robust Dialog State Tracking using Delexicalised Recurrent Neural Networks and Unsupervised Adaptation. In *Proceedings of IEEE SLT*.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014d. Word-Based Dialog State Tracking with Recurrent Neural Networks. In *Proceedings of SIGDIAL*.

Youngsoo Jang, Jiyeon Ham, Byung-Jun Lee, Youngjae Chang, and Kee-Eung Kim. 2016. Neural dialog state tracker for large ontologies by attention mechanism. In *Proceedings of IEEE SLT*.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A Convolutional Neural Network for Modelling Sentences. In *Proceedings of ACL*.

Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of EMNLP*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of ICLR*.

Byung-Jun Lee and Kee-Eung Kim. 2016. Dialog History Construction with Long-Short Term Memory for Robust Generative Dialog State Tracking. *Dialogue & Discourse* 7(3):47–64.

Bing Liu and Ian Lane. 2016a. Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling. In *Proceedings of Interspeech*.

Bing Liu and Ian Lane. 2016b. Joint Online Spoken Language Understanding and Language Modeling with Recurrent Neural Networks. In *Proceedings of SIGDIAL*.

Fei Liu and Julien Perez. 2017. Gated End-to-End Memory Networks. In *Proceedings of EACL*.

F. Mairesse, M. Gasic, F. Jurcicek, S. Keizer, B. Thomson, K. Yu, and S. Young. 2009. Spoken Language Understanding from Unaligned Data using Discriminative Classification Models. In *Proceedings of ICASSP*.

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

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting Word Vectors to Linguistic Constraints. In *Proceedings of HLT-NAACL*.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2015. Multi-domain Dialog State Tracking using Recurrent Neural Networks. In *Proceedings of ACL*.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of ICML*.

Baolin Peng, Kaisheng Yao, Li Jing, and Kam-Fai Wong. 2015. Recurrent Neural Networks with External Memory for Language Understanding. In *Proceedings of the National CCF Conference on Natural Language Processing and Chinese Computing*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of EMNLP*.

Julien Perez. 2016. Spectral decomposition method of dialog state tracking via collective matrix factorization. *Dialogue & Discourse* 7(3):34–46.

Julien Perez and Fei Liu. 2017. Dialog state tracking, a machine reading approach using Memory Network. In *Proceedings of EACL*.

Christian Raymond and Giuseppe Ricardi. 2007. Generative and discriminative algorithms for spoken language understanding. In *Proceedings of Interspeech*.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomas Kocisky, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *ICLR*.

Iman Saleh, Shafiq Joty, Lluís Màrquez, Alessandro Moschitti, Preslav Nakov, Scott Cyphers, and Jim Glass. 2014. A study of using syntactic and semantic structures for concept segmentation and labeling. In *Proceedings of COLING*.

Hongjie Shi, Takashi Ushio, Mitsuru Endo, Katsuyoshi Yamagami, and Noriaki Horii. 2016. Convolutional Neural Networks for Multi-topic Dialog State Tracking. In *Proceedings of IWSDS*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* .

Pei-Hao Su, Milica Gašić, Nikola Mrkšić, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016a. Continuously learning neural dialogue management. In *arXiv preprint: 1606.02689*.

Pei-Hao Su, Milica Gašić, Nikola Mrkšić, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016b. On-line active reward learning for policy optimisation in spoken dialogue systems. In *Proceedings of ACL*.

Kai Sun, Lu Chen, Su Zhu, and Kai Yu. 2014. The SJTU System for Dialog State Tracking Challenge 2. In *Proceedings of SIGDIAL*.

Kai Sun, Qizhe Xie, and Kai Yu. 2016. Recurrent Polynomial Network for Dialogue State Tracking. *Dialogue & Discourse* 7(3):65–88.

Blaise Thomson and Steve Young. 2010. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Computer Speech and Language* .

Gokhan Tur, Anoop Deoras, and Dilek Hakkani-Tur. 2013. Semantic Parsing Using Word Confusion Networks With Conditional Random Fields. In *Proceedings of Interspeech*.

Andreas Vlachos and Stephen Clark. 2014. A new corpus and imitation learning framework for context-dependent semantic parsing. *TACL* 2:547–559.

Miroslav Vodolán, Rudolf Kadlec, and Jan Kleindienst. 2017. Hybrid Dialog State Tracker with ASR Features. In *Proceedings of EACL*.

Ngoc Thang Vu, Pankaj Gupta, Heike Adel, and Hinrich Schütze. 2016. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *Proceedings of ICASSP*.

Ivan Vulić, Nikola Mrkšić, Roi Reichart, Diarmuid Ó Séaghdha, Steve Young, and Anna Korhonen. 2017. Morph-fitting: Fine-tuning word vector spaces with simple language-specific rules. In *Proceedings of ACL*.

Wayne Wang. 1994. Extracting Information From Spontaneous Speech. In *Proceedings of Interspeech*.

Zhuoran Wang and Oliver Lemon. 2013. A Simple and Generic Belief Tracking Mechanism for the Dialog State Tracking Challenge: On the believability of observed information. In *Proceedings of SIGDIAL*.

Tsung-Hsien Wen, Milica Gašić, Dongho Kim, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015a. Stochastic Language Generation in Dialogue using Recurrent Neural Networks with Convolutional Sentence Reranking. In *Proceedings of SIGDIAL*.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015b. Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems. In *Proceedings of EMNLP*.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of EACL*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. From paraphrase database to compositional paraphrase model and back. *TACL* 3:345–358.

Jason D. Williams. 2014. Web-style ranking and SLU combination for dialog state tracking. In *Proceedings of SIGDIAL*.

Jason D. Williams, Antoine Raux, and Matthew Henderson. 2016. The Dialog State Tracking Challenge series: A review. *Dialogue & Discourse* 7(3):4–33.

Jason D. Williams, Antoine Raux, Deepak Ramachandran, and Alan W. Black. 2013. The Dialogue State Tracking Challenge. In *Proceedings of SIGDIAL*.

Jason D. Williams and Steve Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Computer Speech and Language* 21:393–422.

Kaisheng Yao, Baolin Peng, Yu Zhang, Dong Yu, Geoffrey Zweig, and Yangyang Shi. 2014. Spoken language understanding using long short-term memory neural networks. In *Proceedings of ASRU*.

Steve Young, Milica Gašić, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The hidden information state model: A practical framework for POMDP-based spoken dialogue management. *Computer Speech and Language* 24:150–174.

Xiaodong Zhang and Houfeng Wang. 2016. A Joint Model of Intent Determination and Slot Filling for Spoken Language Understanding. In *Proceedings of IJCAI*.

Lukas Zilka and Filip Jurcicek. 2015. Incremental LSTM-based dialog state tracker. In *Proceedings of ASRU*.