Fully Statistical Neural Belief Tracking

Nikola Mrkšić[1] and **Ivan Vulić**[1,2]
[1] PolyAI
[2] Language Technology Lab, University of Cambridge
nikola@poly-ai.com ivan@poly-ai.com

Abstract

This paper proposes an improvement to the existing data-driven Neural Belief Tracking (NBT) framework for Dialogue State Tracking (DST). The existing NBT model uses a hand-crafted belief state update mechanism which involves an expensive manual retuning step whenever the model is deployed to a new dialogue domain. We show that this update mechanism can be *learned* jointly with the semantic decoding and context modelling parts of the NBT model, eliminating the last rule-based module from this DST framework. We propose two different statistical update mechanisms and show that dialogue dynamics can be modelled with a very small number of additional model parameters. In our DST evaluation over three languages, we show that this model achieves competitive performance and provides a robust framework for building resource-light DST models.

1 Introduction

The problem of language understanding permeates the deployment of statistical dialogue systems. These systems rely on dialogue state tracking (DST) modules to model the user's intent at any point of an ongoing conversation (Young, 2010). In turn, DST models rely on domain-specific Spoken Language Understanding (SLU) modules to extract turn-level user goals, which are then incorporated into the *belief state*, the system's internal probability distribution over possible dialogue states.

The dialogue states are defined by the domain-specific *ontology*: it enumerates the constraints the users can express using a collection of slots (e.g. *price range*) and their slot values (e.g. *cheap, expensive* for the aforementioned slots). The belief state is used by the downstream dialogue management component to choose the next system response (Su et al., 2016, 2017).

A large number of DST models (Wang and Lemon, 2013; Sun et al., 2016; Liu and Perez, 2017; Vodolán et al., 2017, *inter alia*) treat SLU as a separate problem: the detached SLU modules are a dependency for such systems as they require large amounts of annotated training data. Moreover, recent research has demonstrated that systems which treat SLU and DST as a single problem have proven superior to those which decouple them (Williams et al., 2016). Delexicalisation-based models, such as the one proposed by (Henderson et al., 2014a,b) offer unparalleled generalisation capability.

These models use *exact matching* to replace occurrences of slot names and values with generic tags, allowing them to share parameters across all slot values. This allows them to deal with slot values not seen during training. However, their downside is shifting the problem of dealing with linguistic variation back to the system designers, who have to craft *semantic lexicons* to specify rephrasings for ontology values. Examples of such rephrasings are [*cheaper*, *affordable*, *cheaply*] for slot-value pair FOOD=CHEAP, or [*with internet*, *has internet*] for HAS INTERNET=TRUE. The use of such lexicons has a profound effect on DST performance (Mrkšić et al., 2016). Moreover, such lexicons introduce a design barrier for deploying these models to large real-world dialogue domains and other languages.

The Neural Belief Tracker (NBT) framework (Mrkšić et al., 2017a) is a recent attempt to overcome these obstacles by using dense word embeddings in place of traditional n-gram features. By making use of semantic relations embedded in the vector spaces, the NBT achieves DST performance competitive to lexicon-supplemented delexicalisation-based models without relying on any hand-crafted resources. Moreover, the NBT

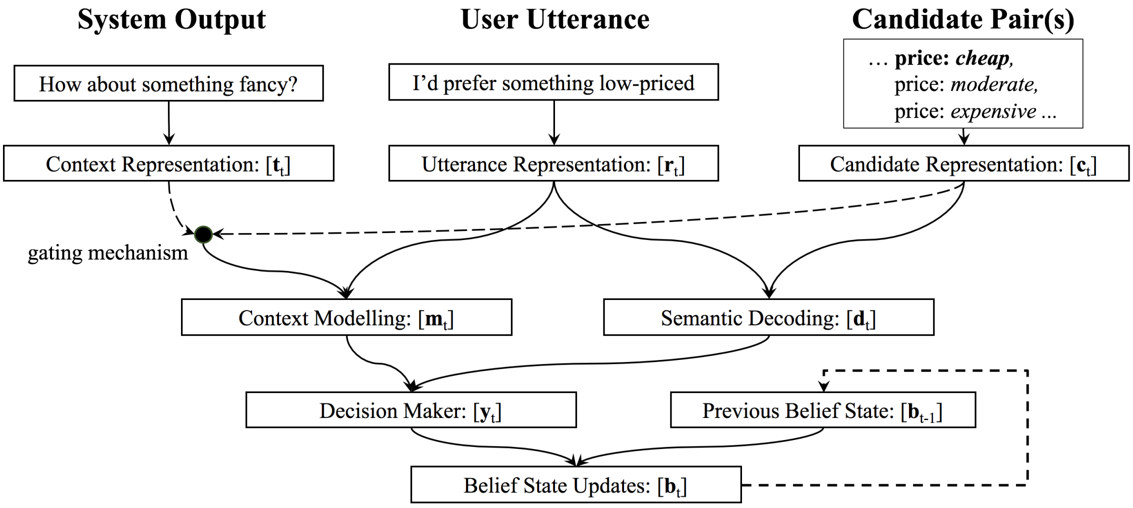
System Output
User Utterance
Candidate Pair(s)
How about something fancy?
I'd prefer something low-priced
... price: cheap, price: moderate, price: expensive ...
Context Representation: [t_t]
Utterance Representation: [r_t]
Candidate Representation: [c_t]
gating mechanism
Context Modelling: [m_t]
Semantic Decoding: [d_t]
Decision Maker: [y_t]
Previous Belief State: [b_t-1]
Belief State Updates: [b_t]

Figure 1: The architecture of the fully statistical neural belief tracker. Belief state updates are not rule-based but learned jointly with the semantic decoding and context modelling parts of the NBT model.

framework enables deployment and bootstrapping of DST models for languages other than English (Mrkšić et al., 2017b). As shown by Vulić et al. (2017), phenomena such as morphology make DST a substantially harder problem in linguistically richer languages such as Italian and German.

The NBT models decompose the (per-slot) multi-class value prediction problem into many binary ones: they iterate through all slot values defined by the ontology and decide which ones have just been expressed by the user. To differentiate between slots, they take as input the word vector of the slot value that it is making a decision about. In doing that, the previous belief state is discarded. However, the previous state may contain pertinent information for making the turn-level decision.

Contribution In this work, we show that cross-turn dependencies can be learned automatically: this eliminates the rule-based NBT component and effectively yields a fully statistical dialogue state tracker. Our competitive results on the benchmarking WOZ dataset for three languages indicate that the proposed fully statistical model: **1)** is robust with respect to the input vector space, and **2)** is easily portable and applicable to different languages.

Finally, we make the code of the novel NBT framework publicly available at: *https://github.com/nmrksic/neural-belief-tracker*, in hope of helping researchers to overcome the initial high-cost barrier to using DST as a real-world language understanding task.

2 Methodology

Neural Belief Tracker: Overview The NBT models are implemented as multi-layer neural networks. Their input consists of three components: **1)** the list of vectors for words in the last user utterance; **2)** the word vectors of the slot name and value (e.g. FOOD=INDIAN) that the model is currently making a decision about; and **3)** the word vectors which represent arguments of the preceding system acts.[1] To perform belief state tracking, the NBT model iterates over all candidate slot-value pairs as defined by the ontology, and decides which ones have just been expressed by the user.

The first layer of the NBT (see Figure 1) learns to map these inputs into intermediate distributed representations of: **1)** the current utterance representation $\mathbf{r}$; **2)** the current candidate slot-value pair $\mathbf{c}$; and **3)** the preceding system act $\mathbf{m}$. These representations then interact through the *context modelling* and *semantic decoding* downstream components, and are finally coalesced into the decision about the current slot value pair by the final *binary decision making* module. For full details of this setup, see the original NBT paper (Mrkšić et al., 2017a).

2.1 Statistical Belief State Updates

The NBT framework effectively recasts the per-slot multi-class value prediction problem as multiple

[1] Following Mrkšić et al. (2017a), we also consider only *system requests* and *system confirmations*, which ask the user to specify the value of a given slot (e.g. *'What kind of venue are you looking for?'*), or to confirm whether a certain intent is part of their belief state (*'Are you looking for Chinese food?'*).

binary ones: this enables the model to deal with slot values unseen in the training data. It iterates through all slot values and decides which ones have just been expressed by the user.

In the original NBT framework (Mrkšić et al., 2017a), the model for turn-level prediction is trained using SGD, maximising the accuracy of turn-level slot-value predictions. These predictions take preceding system acts into account, but *not* the previous belief state. Note that these predictions are done *separately for each slot value*.

Problem Definition For any given slot $s \in V_s$, let $\mathbf{b}_s^{t-1}$ be the true belief state at time $t-1$ (this is a vector of length $|V_s|+2$, accounting for all slot values and two special values, *dontcare* and *NONE*). At turn t, let the intermediate representations representing the preceding system acts and the current user utterance be $\mathbf{m}^t$ and $\mathbf{r}^t$. If the model is currently deciding about slot value $v \in V_s$, let the intermediate candidate slot-value representation be $\mathbf{c}_v^t$. The NBT binary-decision making module produces an estimate $\mathbf{y}_{s,v}^t = P(s, v|\mathbf{r^t}, \mathbf{m^t})$. We aim to combine this estimate with the previous belief state estimate for the entire slot s, $\mathbf{b}_s{}^{t-1}$, so that:

$$\mathbf{b}_s^t = \phi(\mathbf{y}_s^t, \mathbf{b}_s^{t-1}) \tag{1}$$

where $\mathbf{y}_s^t$ is the vector of probabilities for each of the slot values $v \in V_s$.

Previously: Rule-Based The original NBT framework employs a convoluted programmatic rule-based update which is hand-crafted and cannot be optimised or learned with gradient descent methods. For each slot value pair (s, v), its new probability $\mathbf{b}_{s,v}^t$ is computed as follows:

$$\mathbf{b}_{s,v}^t = \lambda \mathbf{y}_{s,v}^t + (1-\lambda)\mathbf{b}_{s,v}^{t-1} \tag{2}$$

λ is a tunable coefficient which determines the relative weight of the turn-level and previous turns' belief state estimates, and is maximised according to DST performance on a validation set. For slot s, the set of its *detected values* at turn t is then given as follows:

$$V_s^t = \{v \in V_s | \mathbf{b}_{s,v}^t \geq 0.5\} \tag{3}$$

For informables (i.e., goal-tracking slots), which unlike requestable slots require belief tracking across turns, if $V_s^t \neq \emptyset$ the value in V_s^t with the highest probability is selected as the current goal. This effectively means that the value with the highest probabilities $\mathbf{b}_{s,v}^t$ at turn t is then chosen as the new goal value, but *only* if its new probability $\mathbf{b}_{s,v}^t$ is greater than 0.5. If no value has probability greater than 0.5, the predicted goal value stays the same as the one predicted in the previous turn - even if its probability $\mathbf{b}_{s,v}^t$ is now less than 0.5.

In the rule-based method, tuning the hyper-parameter λ adjusts how likely any predicted value is to override previously predicted values. However, the "belief state" produced in this manner is not a valid probability distribution. It just predicts the top value using an ad-hoc rule that was empirically verified by Mrkšić et al. (2017a).[2]

This rule-based approach comes at a cost: the NBT framework with such updates is little more than an SLU decoder capable of modelling the preceding system acts. Its parameters do not learn to handle the previous belief state, which is essential for probabilistic modelling in POMDP-based dialogue systems (Young et al., 2010; Thomson and Young, 2010). We now show two update mechanisms that extend the NBT framework to (learn to) perform statistical belief state updates.

1. One-Step Markovian Update To stay in line with the NBT paradigm, the criteria for the belief state update mechanism ϕ from Eq. (1) are: **1)** it is a differentiable function that can be backpropagated during NBT training; and **2)** it produces a valid probability distribution $\mathbf{b}_s^t$ as output. Figure 1 shows our fully statistical NBT architecture.

The first learned statistical update mechanism, termed One-Step Markovian Update, combines the previous belief state $\mathbf{b}_s^{t-1}$ and the current turn-level estimate $\mathbf{y}_s^t$ using a one-step belief state update:

$$\mathbf{b}_s^t = softmax\left(W_{curr}\mathbf{y}_s^t + W_{past}\mathbf{b}_s^{t-1}\right) \tag{4}$$

W_{curr} and W_{past} are matrices which learn to combine the two signals into a new belief state. This variant violates the NBT design paradigm: each row of the two matrices learns to operate over *specific* slot values.[3] Even though turn-level NBT

[2] We have also experimented with a simple model which tunes the hyper-parameter λ during training without the remaining decision logic at each turn t (see Eq. (3)). The belief state update is performed as follows: $\mathbf{b}_s^t = \lambda \mathbf{y}_s^t + (1-\lambda)\mathbf{b}_s^{t-1}$. We note that this simplistic update mechanism performs poorly in practice, with joint goal accuracy on the English DST task in the 0.22-0.25 interval (compare it to the results from Table 1).

[3] This means the model will not learn to predict or maintain slot values as part of the belief state if it has not encountered these values during training.

output $\mathbf{y}_s^t$ may contain the right prediction, the parameters of the corresponding row in W_{curr} will not be trained to update the belief state, since its parameters (for the given value) will not have been updated during training. Similarly, the same row in W_{past} will not learn to maintain the given slot value as part of the belief state.

To overcome the data sparsity and preserve the NBT model's ability to deal with unseen values, one can use the fact that there are fundamentally only two different actions that a belief tracker needs to perform: **1)** maintain the same prediction as in the previous turn; or **2)** update the prediction given strong indication that a new slot value has been expressed. To facilitate transfer learning, the second update variant introduces additional constraints for the one-step belief state update.

2. Constrained Markovian Update This variant constrains the two matrices so that each of them contains only two different scalar values. The first one populates the diagonal elements, and the other one is replicated for all off-diagonal elements:

$$W_{curr,i,j} = \begin{cases} a_{curr}, & \text{if } i = j \\ b_{curr}, & \text{otherwise} \end{cases} \tag{5}$$

$$W_{past,i,j} = \begin{cases} a_{past}, & \text{if } i = j \\ b_{past}, & \text{otherwise} \end{cases} \tag{6}$$

where the four scalar values are learned jointly with other NBT parameters. The diagonal values learn the relative importance of propagating the previous value (a_{past}), or of accepting a newly detected value (a_{curr}). The off-diagonal elements learn how turn-level signals (b_{curr}) or past probabilities for other values (b_{past}) impact the predictions for the current belief state. The parameters acting over all slot values are in this way tied, ensuring that the model can deal with slot values unseen in training.

3 Experimental Setup

Evaluation: Data and Metrics As in prior work the DST evaluation is based on the Wizard-of-Oz (WOZ) v2.0 dataset (Wen et al., 2017; Mrkšić et al., 2017a), comprising 1,200 dialogues split into training (600 dialogues), validation (200), and test data (400). The English data were translated to German and Italian by professional translators (Mrkšić et al., 2017b). In all experiments, we report the standard DST performance measure: *joint goal accuracy*, which is defined as the proportion of dialogue turns where all the user's search goal constraints were correctly identified. Finally, all reported scores are averages over 5 NBT training runs.

	English WOZ 2.0	
Model Variant	GLOVE (DIST)	PARAGRAM-SL999
Rule-Based	80.1	84.2
1. One-Step	80.8	82.1
2. Constrained	**81.8**	**84.8**

Table 1: The English DST performance (*joint goal accuracy*) with standard input word vectors (§3).

Training Setup We compare three belief state update mechanisms (rule-based vs. two statistical ones) fixing all other NBT components as suggested by Mrkšić et al. (2017a): the better-performing NBT-CNN variant is used, trained by Adam (Kingma and Ba, 2015) with dropout (Srivastava et al., 2014) of 0.5, gradient clipping, batch size of 256, and 400 epochs. All model hyperparameters were tuned on the validation sets.

Word Vectors To test the model's robustness, we use a variety of standard word vectors from prior work. For English, following Mrkšić et al. (2017a) we use **1)** distributional GLOVE vectors (Pennington et al., 2014), and **2)** specialised PARAGRAM-SL999 vectors (Wieting et al., 2015), obtained by injecting similarity constraints from the Paraphrase Database (Pavlick et al., 2015) into GLOVE.

For Italian and German, we compare to the work of Vulić et al. (2017), who report state-of-the-art DST scores on the Italian and German WOZ 2.0 datasets. In this experiment, we train the models using distributional skip-gram vectors with a large vocabulary (labelled DIST in Table 2). Subsequently, we compare them to models trained using word vectors specialised using similarity constraints derived from language-specific morphological rules (labelled SPEC in Table 2).

4 Results and Discussion

Table 1 compares the two variants of the statistical update. The Constrained Markovian Update is the better of the two learned updates, despite using only four parameters to model dialogue dynamics (rather than $O(V^2)$, V being the slot value count). This shows that the ability to generalise to unseen slot values matters more than the ability to model value-specific behaviour. In fact, combining the two updates led to no performance gains over the

	Italian		German	
	DIST	SPEC	DIST	SPEC
Rule-Based Update	**74.2**	76.0	60.6	66.3
Learned Update	73.7	**76.1**	**61.5**	**68.1**

Table 2: DST performance on Italian and German. Only results with the better scoring learned Constrained Markovian Update are reported.

stand-alone Constrained Markovian update.

Table 2 investigates the portability of this model to other languages. The statistical update shows comparable performance to the rule-based one, outperforming it in three out of four experiments. In fact, our model trained using the specialised word vectors sets the new state-of-the-art performance for English, Italian and German WOZ 2.0 datasets. This supports our claim that eliminating the hand-tuned rule-based update makes the NBT model more stable and better suited to deployment across different dialogue domains and languages.

DST as Downstream Evaluation All of the experiments show that the use of semantically specialised vectors benefits DST performance. The scale of these gains is robust across all experiments, regardless of language or the employed belief state update mechanism. So far, it has been hard to use the DST task as a proxy for measuring the correlation between word vectors' intrinsic performance (in tasks like SimLex-999 (Hill et al., 2015)) and their usefulness for downstream language understanding tasks. Having eliminated the rule-based update from the NBT model, we make our evaluation framework publicly available in hope that DST performance can serve as a useful tool for measuring the correlation between intrinsic and extrinsic performance of word vector collections.

5 Conclusion

This paper proposed an extension to the Neural Belief Tracking (NBT) model for Dialogue State Tracking (DST) (Mrkšić et al., 2017a). In the previous NBT model, system designers have to tune the *belief state update* mechanism manually whenever the model is deployed to new dialogue domains. On the other hand, the proposed model *learns* to update the belief state automatically, relying on no domain-specific validation sets to optimise DST performance. Our model outperforms the existing NBT model, setting the new state-of-the-art performance for the Multilingual WOZ 2.0 dataset across all three languages. We make the proposed framework publicly available in hope of providing a robust tool for exploring the DST task for the wider NLP community.

Acknowledgments

We thank the anonymous reviewers for their helpful and insightful suggestions. We are also grateful to Diarmuid Ó Séaghdha and Steve Young for many fruitful discussions.

References

Matthew Henderson, Blaise Thomson, and Steve Young. 2014a. Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In *Proceedings of IEEE SLT*, pages 360–365.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014b. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of SIGDIAL*, pages 292–299.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR (Conference Track)*.

Fei Liu and Julien Perez. 2017. Gated end-to-end memory networks. In *Proceedings of EACL*, pages 1–10.

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 NAACL-HLT*, pages 142–148.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Tsung-Hsien Wen, and Steve Young. 2017a. Neural Belief Tracker: Data-driven dialogue state tracking. In *Proceedings of ACL*, pages 1777–1788.

Nikola Mrkšić, Ivan Vulić, Diarmuid Ó Séaghdha, Ira Leviant, Roi Reichart, Milica Gašić, Anna Korhonen, and Steve Young. 2017b. Semantic specialisation of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the ACL*, pages 314–325.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of ACL*, pages 425–430.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP*, pages 1532–1543.

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*, 15:1929–1958.

Pei-Hao Su, Paweł Budzianowski, Stefan Ultes, Milica Gašić, and Steve Young. 2017. Sample-efficient actor-critic reinforcement learning with supervised data for dialogue management. pages 147–157.

Pei-Hao Su, Milica Gašić, Nikola Mrkšić, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. On-line active reward learning for policy optimisation in spoken dialogue systems. In *Proceedings of ACL*, pages 2431–2441.

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 & Language*, 24(4):562–588.

Miroslav Vodolán, Rudolf Kadlec, and Jan Kleindienst. 2017. Hybrid dialog state tracker with ASR features. In *Proceedings of EACL*, pages 205–210.

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*, pages 56–68.

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, 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*, pages 438–449.

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

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

Steve Young. 2010. Cognitive user interfaces. *IEEE Signal Processing Magazine*.

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 & Language*, 24(2):150–174.