Uncertainty Measures in Neural Belief Tracking and the Effects on Dialogue Policy Performance

Carel van Niekerk[1], **Andrey Malinin**[2], **Christian Geishauser**[1], **Michael Heck**[1]
Hsien-chin Lin[1], **Nurul Lubis**[1], **Shutong Feng**[1] **and Milica Gašić**[1]
[1]Heinrich Heine Universität Düsseldorf, Düsseldorf, Germany
[2]Yandex Research and HSE University, Moscow, Russia
[1]{niekerk,geishaus,heckmi,linh,lubis,shutong.feng,gasic}@hhu.de
[2]am969@yandex-team.ru

Abstract

The ability to identify and resolve uncertainty is crucial for the robustness of a dialogue system. Indeed, this has been confirmed empirically on systems that utilise Bayesian approaches to dialogue belief tracking. However, such systems consider only confidence estimates and have difficulty scaling to more complex settings. Neural dialogue systems, on the other hand, rarely take uncertainties into account. They are therefore overconfident in their decisions and less robust. Moreover, the performance of the tracking task is often evaluated in isolation, without consideration of its effect on the downstream policy optimisation. We propose the use of different uncertainty measures in neural belief tracking. The effects of these measures on the downstream task of policy optimisation are evaluated by adding selected measures of uncertainty to the feature space of the policy and training policies through interaction with a user simulator. Both human and simulated user results show that incorporating these measures leads to improvements both of the performance and of the robustness of the downstream dialogue policy. This highlights the importance of developing neural dialogue belief trackers that take uncertainty into account.

1 Introduction

In task-oriented dialogue, the system aims to assist the user in obtaining information. This is achieved through a series of interactions between the user and the system. As the conversation progresses, it is the role of the *dialogue state tracking* module to track the state of the conversation. For example, in a restaurant recommendation system, the state would include the information about the cuisine of the desired restaurant, its area as well as the price range that the user has in mind. It is crucial that this state contains all information necessary for the *dialogue policy* to make an informed decision for the next action (Young et al., 2007). Policy training optimises decision making in order to complete dialogues successfully.

It has been proposed within the partially observable Markov decision process (POMDP) approach to dialogue modelling to track the distribution over all possible dialogue states, the *belief state*, instead of a single most-likely candidate. This approach successfully integrates uncertainty to achieve robustness (Williams and Young, 2007; Thomson and Young, 2010; Young et al., 2016, 2007). However, such systems do not scale well to complex multi-domain dialogues. On the other hand, discriminative neural approaches to dialogue tracking achieve state-of-the-art performance in the state tracking task. Nevertheless, the state-of-the-art goal accuracy on the popular MultiWOZ (Budzianowski et al., 2018) multi-domain benchmark is currently only at 60% (Heck et al., 2020; Li et al., 2020a). In other words, even the best neural dialogue state trackers at present incorrectly predict the state of the conversation in 40% of the turns. What is particularly problematic is that these models are fully confident about their incorrect predictions.

Unlike neural dialogue state trackers, which predict a single best dialogue state, neural belief trackers produce a belief state (Williams and Young, 2007; Henderson et al., 2013). State-of-the-art neural belief trackers, however, achieve an even lower goal accuracy of approximately 50% (van Niekerk et al., 2020; Lee et al., 2019), making the more accurate state trackers a preferred approach. High-performing state trackers typically rely on span-prediction approaches, which are unable to produce a distribution over all possible states as they extract information directly from the dialogue.

Ensembles of models are known to yield improved predictive performance as well as a calibrated and rich set of uncertainty estimates (Malinin, 2019; Gal, 2016). Unfortunately, ensemble generation and, especially, inference come at a high computational and memory cost which may be pro-

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 7901–7914
November 7–11, 2021.

hibitive. While standard ensemble distillation (Hinton et al., 2015) can be used to compress an ensemble into a single model, information about ensemble diversity, and therefore several uncertainty measures, is lost. Recently Malinin et al. (2019) and Ryabinin et al. (2021) proposed *ensemble distribution distillation* (EnD^2) - an approach to distill an ensemble into a single model which preserves both the ensemble's improved performance and full set of uncertainty measures at low inference cost.

In this work we use EnD^2 to distill an ensemble of neural belief trackers into a single model and incorporate additional uncertainty measures, namely confidence scores, total uncertainty (entropy) and knowledge uncertainty (mutual information), into the belief state of the neural dialogue system. This yields an uncertainty-aware neural belief tracker and allows downstream dialogue policy models to use this information to resolve confusion. To our knowledge, ensemble distillation, especially ensemble distribution distillation, and the derived uncertainty estimates, have not been examined for belief state estimation or *any* downstream tasks.

We make the following contributions:

1. We present SetSUMBT, a modified SUMBT belief tracking model, which incorporates set similarity for accurate state predictions and produces components essential for policy optimisation.
2. We deploy ensemble distribution distillation to obtain well-calibrated, rich estimates of uncertainty in the dialogue belief tracker. The resulting model produces state-of-the-art results in terms of calibration measures.
3. We demonstrate the effect of adding additional uncertainty measures in the belief state on the downstream dialogue policy models and confirm the effectiveness of these measures both in a simulated environment and in a human trial.

2 Background

2.1 Dialogue Belief Tracking

In statistical approaches to dialogue, one can view the dialogue as a Markov decision process (MDP) (Levin et al., 1998). This MDP maintains a Markov *dialogue state* in each turn and chooses its next *action* based on this state.

Alternatively, we can model the dialogue state as a latent variable, maintaining a *belief state* at each turn, as in partially observable Markov decision processes (POMDPs) (Williams and Young, 2007; Thomson and Young, 2010). While attractive in theory, the POMDP model is computationally expensive in practice. Although there are practical implementations, they are limited to single-domain dialogues and their performance fall short of discriminative statistical belief trackers (Williams, 2012). The inherent problem lies in the generative nature of POMDP trackers where the state generates noisy observations. This becomes an issue for instance when the user wants to change the goal of a conversation, e.g., the user wants an Italian instead of a French restaurant. Henderson (2015) has shown empirically that discriminative models model a change in user goal more accurately.

In discriminative approaches, the state depends on the observation, making it easier for the system to identify a change of the user goal. Traditional discriminative approaches suffer from low robustness, as they depend on static semantic dictionaries for feature extraction (Henderson et al., 2014; Mrkšić et al., 2017b). Integrated approaches on the other hand utilise learned token vector representations, leading to more robust state trackers (Mrkšić et al., 2017a; Ramadan et al., 2018; Lee et al., 2019). However, highly over-parameterised models, such as neural networks – when trained via maximum-likelihood on finite data – often yield miscalibrated, over-confident predictions, placing *all* probability mass on a single outcome (Pleiss et al., 2017). Consequently, belief tracking is reduced to state tracking, losing the benefits of uncertainty management. State-of-the-art approaches to dialogue state tracking redefine the problem as a span-prediction task. These models extract the values directly from the dialogue context (Chao and Lane, 2019; Zhang et al., 2020; Heck et al., 2020) and manage to achieve state-of-the-art results on MultiWOZ (Budzianowski et al., 2018; Eric et al., 2020). Span-prediction models at present do not produce probability distributions, so additional work is needed to apply our proposed uncertainty framework to them. Neural belief and state trackers rarely model the correlation between domain-slot pairs, except for works by Hu et al. (2020) and Ye et al. (2021). Due to scalability issues we do not include these approaches in our investigation. We therefore consider the slot-utterance matching belief tracker (SUMBT) (Lee et al., 2019) a better starting point, as it is readily able to produce a belief state distribution.

In theory, well-calibrated belief trackers have an inherent advantage over state tracking, producing uncertainty estimates that lead to more robust downstream policy performance. This raises the question: Is it possible to instil well-calibrated uncertainty estimates in neural belief trackers? And if so, do these estimates have a positive effect on the downstream policy optimisation in practice?

We believe SUMBT is a fitting approach to investigate these questions, as it has been shown that an ensemble of SUMBT models can achieve state-of-the-art goal L2-Error when trained using specialised loss functions aiming at inducing uncertainty in the output (van Niekerk et al., 2020).

2.2 Ensemble-based Uncertainty Estimation

Consider a classification problem with a set of features $\boldsymbol{x}$, and outcomes $y \in \{\omega_1, \omega_2, ..., \omega_K\}$. In dialogue state tracking, $\boldsymbol{x}$ would be features of the input to the tracker and y would be a dialogue state. Given an ensemble of M models $\left\{\mathtt{P}(y|\boldsymbol{x}, \boldsymbol{\theta}^{(m)})\right\}_{m=1}^{M}$, the *predictive posterior* is obtained as follows:

$$\mathtt{P}(y|\boldsymbol{x}, \mathcal{D}) = \sum_{m=1}^{M} \frac{\mathtt{P}(y|\boldsymbol{x}, \boldsymbol{\theta}^{(m)})}{M} =: \sum_{m=1}^{M} \frac{\boldsymbol{\pi}^{(m)}}{M} \quad (1)$$

Predictions made using the predictive posterior are often better than those of individual models. The entropy $\mathcal{H}[]$ of the predictive posterior is an estimate of *total uncertainty*. Ensembles allow decomposing *total uncertainty* into *data* and *knowledge uncertainty* by considering measures of *ensemble diversity*. *Data uncertainty* is the uncertainty due to noise, ambiguity and class overlap in the data. *Knowledge uncertainty* is uncertainty due to a lack of knowledge of the model about a *test data* (Malinin, 2019; Gal, 2016) — ie, uncertainty due to unfamiliar, anomalous or atypical inputs. Ideally, ensembles should yield *consistent* predictions on data similar to the training data and *diverse* predictions on data which is significantly different from the training data. Thus measures of ensemble diversity yield estimates of *knowledge uncertainty*[1]. These quantities are obtained via the mutual information $\mathcal{I}[y, \boldsymbol{\theta}]$ between predictions and model parameters. The quantity in the Equation 2 is a measure of *ensemble diversity*, and therefore, *knowledge uncertainty*. This quantity is the difference between the entropy of the predictive posterior (total uncertainty) and the average entropy of each model in the ensemble (data uncertainty).

$$\underbrace{\mathcal{I}[y, \boldsymbol{\theta}|\boldsymbol{x}, \mathcal{D}]}_{\text{Knowledge unc.}} = \underbrace{\mathcal{H}\big[\mathtt{P}(y|\boldsymbol{x}, \mathcal{D})\big]}_{\text{Total uncertainty}} - \underbrace{\sum_{m=1}^{M} \frac{\mathcal{H}\big[\mathtt{P}(y|\boldsymbol{x}, \boldsymbol{\theta}^{(m)})\big]}{M}}_{\text{Data uncertainty}} \quad (2)$$

2.3 Ensemble Distillation

While ensembles provide improved predictive performance and a rich set of uncertainty measures, their practical application is limited by their inference-time computational cost. Ensemble distillation (EnD) (Hinton et al., 2015) can be used to compress an ensemble into a single student model (with parameters $\boldsymbol{\phi}$) by minimising the Kullback-Leibler (KL) divergence between the ensemble predictive posterior and the distilled model predictive posterior, significantly reducing the inference cost. Unfortunately, a significant drawback of this method is that information about *ensemble diversity*, and therefore *knowledge uncertainty*, is lost in the process. Recently, Malinin et al. (2019) proposed *ensemble distribution distillation* (EnD^2) as an approach to distill an ensemble into a single *prior network* model (Malinin and Gales, 2018), such that the model retains information about ensemble diversity. Prior networks yield a higher-order *Dirichlet distribution* over categorical output distributions $\boldsymbol{\pi}$ and thereby *emulate* ensembles, whose output distributions can be seen as samples from a higher-order distribution[2]. Formally, a prior network is defined as follows:

$$\begin{aligned} &\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x}; \boldsymbol{\phi}) = \mathtt{Dir}(\boldsymbol{\pi}|\boldsymbol{\alpha}), \ \boldsymbol{\alpha} = e^{\boldsymbol{z}} \\ &\boldsymbol{z} = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{\phi}), \ \alpha_k > 0, \ \alpha_0 = \sum_{k=1}^{K} \alpha_k, \end{aligned} \quad (3)$$

where $\mathtt{Dir}(\cdot|\boldsymbol{\alpha})$ is a Dirichlet distribution with concentration parameters $\boldsymbol{\alpha}$, and $\boldsymbol{f}(\cdot; \boldsymbol{\phi})$ is a learned function which yields the logits $\boldsymbol{z}$. The predictive posterior can be obtained in closed form though marginalisation over $\boldsymbol{\pi}$, thereby emulating (1). This yields a softmax output function:

$$\begin{aligned} \mathtt{P}(y = \omega_k|\boldsymbol{x}; \boldsymbol{\phi}) &= \mathbb{E}_{\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x}; \boldsymbol{\phi})}[\mathtt{P}(y = \omega_k|\boldsymbol{\pi})] \\ &= \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}}. \end{aligned} \quad (4)$$

[1] In-depth overviews of ensemble methods are available in Malinin (2019); Gal and Ghahramani (2016); Ashukha et al. (2020); Ovadia et al. (2019).

[2] $\boldsymbol{\pi} = [\mathtt{P}(y = \omega_1|\boldsymbol{x}), \cdots, \mathtt{P}(y = \omega_K|\boldsymbol{x})]^{\mathtt{T}}$.

Closed form estimates of all uncertainty measures are obtained via Eq. (5), which emulates the same underlying mechanics as Eq. (2), as follows (Malinin, 2019):

$$\underbrace{\mathcal{I}[y, \boldsymbol{\pi}|\boldsymbol{x}; \boldsymbol{\phi}]}_{\text{Knowledge unc.}} = \underbrace{\mathcal{H}\left[\mathtt{P}(y|\boldsymbol{x}; \boldsymbol{\phi})\right]}_{\text{Total uncertainty}} - \underbrace{\mathbb{E}_{\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x};\boldsymbol{\phi})}\left[\mathcal{H}[\mathtt{P}(y|\boldsymbol{\pi})]\right]}_{\text{Data uncertainty}}. \quad (5)$$

Originally, Malinin et al. (2019) implemented EnD^2 on the CIFAR10, CIFAR100 and TinyImageNet datasets. However, Ryabinin et al. (2021) found scaling to tasks with many classes challenging using the original Dirichlet Negative log-likelihood criterion. They analysed this scaling problem and proposed to a new loss function, which minimises the *reverse KL-divergence* between the model and an intermediate *proxy Dirichlet* target derived from the ensemble. This loss function was shown to enable EnD^2 on tasks with arbitrary numbers of classes. In this work we use this improved loss function, as detailed in the Appendix Section B.2.

2.4 Policy Optimisation

In each turn of dialogue, the dialogue policy selects an action to take in order to successfully complete the dialogue. The input to the policy is constructed using the output of the belief state tracker, thus being directly impacted by its richness.

Optimising dialogue policies within the original POMDP framework is not practical for most cases. Therefore, the POMDP is viewed as a continuous MDP whose state space is the belief space. This state space can be discretised, so that tabular reinforcement learning (RL) algorithms can be applied (Gašić et al., 2008; Thomson et al., 2010). Gaussian process RL can be applied directly on the original belief space (Gašić and Young, 2014). This is also possible using neural approaches with less computational effort (Jurčíček et al., 2011; Weisz et al., 2018; Chen et al., 2020). Current state-of-the-art RL algorithms for multi-domain dialogue management (Takanobu et al., 2019; Li et al., 2020b) utilise proximal policy optimisation (Schulman et al., 2017) operating on single best dialogue state.

3 Effects of Uncertainty on Downstream Tasks

We take the following steps in order to examine the effects of the additional uncertainty measures in the dialogue belief state:

1. Modify the original SUMBT model (Lee et al., 2019) to arrive at a competitive baseline. We call this model SetSUMBT.
2. Produce ensembles of SetSUMBT following the work of van Niekerk et al. (2020).
3. Apply EnD and EnD^2 as introduced in Section 2.3.
4. Apply policy optimisation that uses belief states from distilled models.

3.1 Neural Belief Tracking Model

We propose a neural belief tracker which one can easily incorporate in a full dialogue system pipeline. We base our tracker on the slot-utterance matching belief tracker (SUMBT) (Lee et al., 2019), but we make two important changes. First, we ensure our tracker is fully in line with the requirements of the hidden information state (HIS) model for dialogue management (Young et al., 2007) by adding user action predictions to our tracker. These are not produced by the SUMBT model and nor by other available neural trackers. However, they are essential for integration into a full dialogue system. Second, in order to improve the understanding ability of the model, we utilise a set of concept description embeddings rather than a single embedding for semantic concepts. We use this set of embeddings for information extraction and prediction, hence we call our model SetSUMBT. In this section we describe each component in detail, also depicted in Figure 1.

Slot-utterance matching The slot-utterance matching (SUM) component performs the role of language understanding in the SUMBT architecture. The SUM multi-head attention mechanism (Vaswani et al., 2017) attends to the relevant information in the current turn for a specific domain-slot pair. In the process of slot-utterance matching, SUMBT utilises BERT's (Devlin et al., 2019) `[CLS]` sequence embedding to represent the semantic concepts in the model ontology. Instead of using the single `[CLS]` embedding, we make use of the sequence of embeddings for the domain-slot description. We choose to make this expansion, as approaches which utilise a sequence of embeddings outperform approaches based on a single embedding in various natural language processing tasks (Poerner et al., 2020; Choi et al., 2021). We further use RoBERTa as a feature extractor (Liu et al., 2019).

Dialogue context tracking The first of the three components of the HIS model is a representation of the dialogue context (history). In the SUMBT approach, a gated-recurrent unit mechanism tracks the most important information during a dialogue. The resulting context conditioned representations for the domain-slot pairs contain the relevant information from the dialogue history. Similar to the alteration in the SUM component, we represent the dialogue context as a sequence of representations. This sequence, $\boldsymbol{C}_t^s$, represents the dialogue context for domain-slot pair s across turns 1 to t, while it dimension being independent of t. Besides the above modification, we add a further step where we reduce this sequence of context representations to a single representation $\hat{\mathbf{y}}_t^s$. We do this reduction using a learned convolutional pooler, which we call the *Set Pooler*. See Appendix Section C for more details regarding the implementation.

User goal prediction The second component of the HIS model is the user goal. This is typically the only component that neural tracing models explicitly model as a set of domain-slot-value pairs. Here, we follow the matching network approach (Vinyals et al., 2016) utilised by SUMBT, where the predictive distribution is based on the similarity between the dialogue context and the value candidates. To obtain the similarity between the dialogue context and a value candidate we make use of cosine similarity, $S_{\cos}(\cdot,\cdot)$. Based on these similarity scores, we produce a predictive distribution, Equation 6, for the value of domain-slot pair s at turn t v_t^s, the user and system utterances at turn t, $\boldsymbol{u}_t^{usr}$ and $\boldsymbol{u}_{t-1}^{sys}$, and the dialogue context representations at turn $t-1$ $\boldsymbol{C}_{t-1}^s$. Contrary to the SUMBT approach, each value candidate is represented by the sequence of value description embeddings from a fixed RoBERTa model. The *Set Pooler*, with the same parameters used for pooling context representations, reduces this sequence of value description representations to a representation $\mathbf{y}_v$, for value v.

$$\mathtt{P}\left(v_t^s = v|\boldsymbol{u}_t^{usr}, \boldsymbol{u}_{t-1}^{sys}, \boldsymbol{C}_{t-1}^s\right) = \frac{\exp\left(S_{\cos}\left(\hat{\mathbf{y}}_t^s, \mathbf{y}_v\right)\right)}{\sum_{v'}\exp\left(S_{\cos}\left(\hat{\mathbf{y}}_t^s, \mathbf{y}_{v'}\right)\right)}, \tag{6}$$

User action prediction To be fully in line with the HIS model, we further require the predicted user actions. In order to predict the user actions, we categorise them into general user actions and user request actions. Further, since our system is a multi-domain system, we include the current active domain in the hidden information state of the system.

General user action includes actions such as the user thanking or greeting the system, which do not rely on the dialogue context. Hence, we can infer general user actions from the current user utterance. A user request action is an action indicating that the user is requesting information about an entity. Zhu et al. (2020) shows that simple rule-based estimates of these actions lead to poor downstream policy performance. Hence, we propose predicting this information within the belief tracking model.

Since we can infer the general actions from the current user utterance, we use a single turn representation $\boldsymbol{x}_t^0$ to predict such actions. The single turn representation, $\boldsymbol{x}_t^0$, is the representation for the RoBERTa sequence representation `<s>`, which is equivalent to the BERT `[CLS]` representation. That is:

$$\begin{aligned}\mathtt{P}\left(a_t^{gen} = a|\mathbf{u}_t^{usr}, \mathbf{u}_{t-1}^{sys}\right) = \\ \text{softmax}\left(\boldsymbol{W}^{gen}\mathbf{x}_t^0 + \boldsymbol{b}^{gen}\right),\end{aligned} \tag{7}$$

where $a \in \{\texttt{none}, \texttt{thank\_you}, \texttt{goodbye}\}$.

The more difficult sub-tasks include active user request and active domain prediction. For user request prediction we utilise the dialogue context representation $\hat{\mathbf{y}}_t^s$ for a specific domain-slot pair to predict whether the user has requested information relating to this slot. That is:

$$\begin{aligned}\mathtt{P}\left(r_t^s = 1|\boldsymbol{u}_t^{usr}, \boldsymbol{u}_{t-1}^{sys}, \boldsymbol{C}_{t-1}^s\right) = \\ \text{sigmoid}\left(\boldsymbol{w}^{req}\hat{\mathbf{y}}_t^s + b^{req}\right),\end{aligned} \tag{8}$$

where r_t^s indicates an active request for domain-slot s by the user in turn t.

Last, to predict active domains in the dialogue, we incorporate information relating to all slots associated with a specific domain. We do so by performing mean reduction across the context representations of all the slots associated with a domain. The resulting domain representations are used to predict whether a domain is currently being discussed in the dialogue. That is, for active domain d_t, S_d the set of slots within domain d, and $\boldsymbol{C}_{t-1}^d := \left[\boldsymbol{C}_{t-1}^s\right]_{s\in S_d}$ the set of context representations for all domain-slot pairs in S_d at turn $t-1$, we have the active domain distribution:

$$\begin{aligned}\mathtt{P}\left(d_t = d|\boldsymbol{u}_t^{usr}, \boldsymbol{u}_{t-1}^{sys}, \boldsymbol{C}_{t-1}^d\right) = \\ \text{sigmoid}\left(\boldsymbol{w}^{dom}\frac{1}{|S_d|}\sum_{s\in S_d}\hat{\mathbf{y}}_t^s + b^{dom}\right),\end{aligned} \tag{9}$$

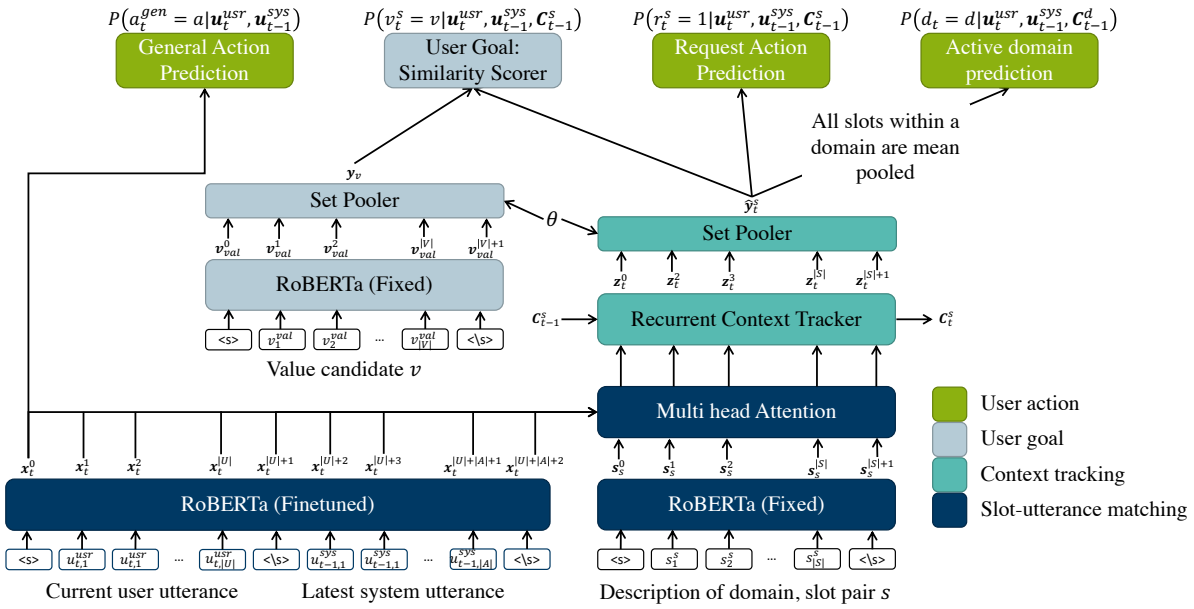

Figure 1: Architecture of our SetSUMBT model, which takes as input the current user utterance, the latest system utterance, and a domain-slot pair description. The model, further, requires a pre-defined set of plausible value candidates for each domain-slot pair. At each turn, we encode the utterances only once, the *Slot-utterance matching* and *Context tracking* components are utilised once for each domain-slot pair. Further, we use the *Set Pooler* once for each domain-slot pair and once for each value candidate. The *Set Pooler* used for pooling value candidate and domain-slot context sequences shares the same parameters $\boldsymbol{\theta}$. SetSUMBT outputs a belief state distribution for the relevant domain-slot pair (*User goal*), a distribution over general actions, and the probability of a user request for the domain-slot pair (*User action*). The model also outputs the probability of an active domain.

Optimisation For each of the four tasks: user goal prediction, general user action prediction, user request action prediction and active domain prediction, the aim of the model is to predict the correct class. To optimise for these objectives, we minimise the following classification loss functions: $\mathcal{L}_{\text{goal}}$, $\mathcal{L}_{\text{general}}$, $\mathcal{L}_{\text{request}}$ and $\mathcal{L}_{\text{domain}}$. During model training we combine four weighted classification objectives:

$$\begin{aligned}\mathcal{L} = \alpha_{\text{goal}}\mathcal{L}_{\text{goal}} + \alpha_{\text{general}}\mathcal{L}_{\text{general}} \\ +\alpha_{\text{request}}\mathcal{L}_{\text{request}} + \alpha_{\text{domain}}\mathcal{L}_{\text{domain}},\end{aligned} \tag{10}$$

where $\alpha_x \in (0, 1]$ is the importance of task x. In this work, we use the label smoothing classification loss for all sub-tasks as it results in better calibrated predictions, as shown by van Niekerk et al. (2020), see details in Section B.1 of the appendix.

3.2 Uncertainty Estimation in SetSUMBT

Similarly to van Niekerk et al. (2020), we construct an ensemble of SetSUMBT models by training each model on one of 10 randomly selected subsets of data. We then distil this ensemble into a single model by adopting ensemble distillation (EnD) and ensemble distribution distillation (EnD^2) as described in Section 2.3. We refer to these distilled versions of the SetSUMBT ensemble as EnD-SetSUMBT and EnD^2-SetSUMBT, respectively.

The SetSUMBT belief tracker tracks the presence and value of each domain-slot pair s as the dialogue progresses. For the sake of scalability of the downstream policy, in the user goal g we do not consider all possible values, but rather the most likely one v^s for every domain-slot pair s and its associated probability, i.e., the confidence score given by $h^g_{t,s}$ summarised in vector $\mathbf{h}^g_t$ for all domain-slot pairs:

$$\begin{aligned} v^s &:= \arg\max_v \mathrm{P}\left(v^s_t = v | \boldsymbol{u}^{usr}_t, \boldsymbol{u}^{sys}_{t-1}, \boldsymbol{C}^s_{t-1}\right), \\ h^g_{t,s} &:= \max_v \mathrm{P}\left(v^s_t = v | \boldsymbol{u}^{usr}_t, \boldsymbol{u}^{sys}_{t-1}, \boldsymbol{C}^s_{t-1}\right), \\ \mathbf{h}^g_t &:= [v^s, h^g_{t,s}]_{\forall s}. \end{aligned} \tag{11}$$

For the EnD-SetSUMBT belief tracker, we can also calculate the total uncertainty for each domain-slot given by the entropy, see Section 2.3. We encode that information in $h^{unc}_{t,s}$ for each domain-slot pair

s and summarise in $\mathbf{h}_t^{unc}$ for all domain-slot pairs:

$$h_{t,s}^{unc} \coloneqq \mathcal{H}\left[\mathtt{P}\left(v_t^s = v | \boldsymbol{u}_t^{usr}, \boldsymbol{u}_{t-1}^{sys}, \boldsymbol{C}_{t-1}^s\right)\right],$$
$$\mathbf{h}_t^{unc} \coloneqq [h_{t,s}^{unc}]_{\forall s}.$$

For the EnD^2-SetSUMBT belief tracker, can further include the knowledge uncertainty for each domain-slot pair s given by the mutual information:

$$h_{t,s}^{unc} \coloneqq \mathcal{I}[v_t^s, \boldsymbol{\pi} | \boldsymbol{u}_t^{usr}, \boldsymbol{u}_{t-1}^{sys}, \boldsymbol{C}_{t-1}^s; \boldsymbol{\phi}],$$

as per Eq. (5) where $\boldsymbol{\pi}$ represents the ensemble distribution and $\boldsymbol{\phi}$ the model parameters.

In addition, all versions of SetSUMBT include the following vectors/variables:
$\mathbf{h}_t^g$ is the estimate of the user goal from Eq. (11),
$\mathbf{h}_t^{usr}$ is the estimate of user actions from Eq. (7-9),
$\mathbf{h}_t^{db}$ is the database search result[3],
$\mathbf{h}_{t-1}^{sys}$ is the system action,
$\mathbf{h}_t^{book}$ is the set of completed bookings,
h_t^{term} indicates the termination of the dialogue.

This results in the following belief state:

$$\mathbf{b}_t = \{\mathbf{h}_t^{usr}, \mathbf{h}_{t-1}^{sys}, \mathbf{h}_t^g, \mathbf{h}_t^{book}, \mathbf{h}_t^{db}, h_t^{term}, \mathbf{h}_t^{unc}\}.$$

For a system without uncertainty, all confidences would be rounded to either 0 or 1 and the belief state would not contain the $\mathbf{h}_t^{unc}$ vector.

3.3 Policy Optimisation as Downstream Task

For our experiments we optimise the dialogue policy operating on the belief state via RL using the PPO algorithm (Schulman et al., 2017). PPO is an on-policy actor-critic algorithm that is widely applied across different reinforcement learning tasks because of its good performance and simplicity. Similarly to Takanobu et al. (2019), we use supervised learning to pretrain the policy before starting the RL training phase. In order to perform supervised learning we need to map the belief states into system actions as they occur in the corpus. These belief states can either be oracle states taken from the corpus or predictions of our belief tracker that takes corpus dialogues as input. We investigate both options for policy training.

[3]Uncertainty is incorporated in the database query vector in the form of confidence thresholds. If the confidence score for a specific constraint is less than a chance prediction then this constraint is ignored during the database query.

Approach	JGA(%)	L2-Error	ECE(%)
SUMBT	46.78	1.1075	25.46
CE-BST	48.71	**1.1042**	10.73
SUMBT+LaRL	51.52	-	-
SetSUMBT	51.11	1.2386	15.13
EnD^2-SetSUMBT	51.22	1.1948	7.09
CE-SetSUMBT	52.04	1.1936	**6.84**
EnD-SetSUMBT	**52.26**	1.1782	7.54

Table 1: Comparison of neural belief tracking approaches on the MultiWOZ 2.1 test set. CE is an ensemble of calibrated models, EnD is ensemble distillation and EnD^2 is ensemble distribution distillation.

4 Experiments

4.1 Neural Belief Tracking Performance

Overall performance Table 1 compares the performance of our proposed SetSUMBT belief tracker to existing approaches, which include SUMBT, the calibrated ensemble belief state tracker (CE-BST) (van Niekerk et al., 2020) and the end to end trained SUMBT+LaRL approach (Lee et al., 2020). We consider the joint goal accuracy (JGA), L2-Error and expected calibration error (ECE). The JGA of a belief tracking model is the percentage of turns for which the model correctly predicted the value for all domain-slot pairs. The L2-Error is the L2-Norm of the difference between the predicted user distribution and the true user goal. Further, the ECE is the average absolute difference between the accuracy and the confidence of a model. In this comparison, we do not consider state tracking approaches, as they do not yield uncertainty estimates. SetSUMBT outperforms SUMBT and SUMBT+LaRL in terms of calibration and accuracy. We name the variants of SetSUMBT as follows: CE-SetSUMBT is a calibrated ensemble of SetSUMBT similar to CE-BST, EnD-SetSUMBT is the distilled SetSUMBT model, and EnD^2-SetSUMBT is the distribution distilled SetSUMBT model.

Runtime efficiency The single instance of the SetSUMBT tracker processes a dialogue turn in approximately 77.768 ms, whereas an ensemble of 10 models processes a turn in approximately 768.025 ms. These processing times are averaged across the 7372 turns in the MultiWOZ test set, see Appendix Section E for more details. The significant increase in processing time for the ensemble of models makes this approach inappropriate for real time interaction with users on a private device.

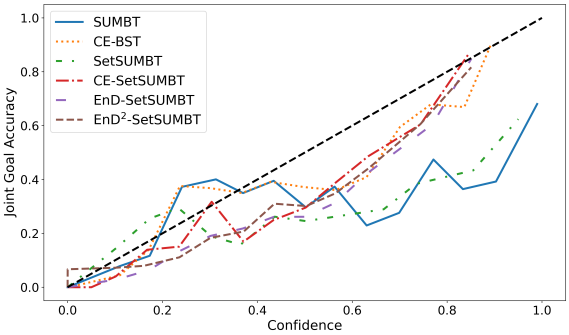

Figure 2: Reliability comparison of a selection of neural belief tracking models.

Belief Tracker	Belief state uncertainty	Success Rate	Reward	Turns
SetSUMBT	None	78.67	46.51	7.49
	Confidence	83.25	52.49	6.80
EnD-SetSUMBT	None	82.25	51.18	7.52
	Confidence	83.75	54.04	**6.46**
	Total	**86.83**	**57.09**	7.35
EnD^2-SetSUMBT	None	83.75	53.00	7.50
	Confidence	84.08	53.15	7.74
	Total	84.83	54.54	7.26
	Knowledge	85.17	54.63	7.57

Table 2: Performance of the systems in the simulated environment. For each setting we have 5 policies initiated with different random seeds, each evaluated with 1000 dialogues and their success rates, reward and number of turns averaged.

Calibration The reliability diagram in Figure 2 illustrates the relationship between the joint goal accuracy and the model confidence. The best calibrated model is the one that is closest to the diagonal, i.e., the one whose confidence for each dialogue state is closest to the achieved accuracy. The best reliability is achieved by CE-BST, and CE-SetSUMBT comes second. Both distillation models (EnD-SetSUMBT and EnD^2-SetSUMBT) do not deviate greatly from CE-SetSUMBT.

4.2 Policy Training on User Simulator

We incorporate SetSUMBT, EnD-SetSUMBT and EnD^2-SetSUMBT within the Convlab2 (Zhu et al., 2020) task-oriented dialogue environment and compare their performance by training policies which take belief states as inputs[4].

To investigate the impact of additional uncertainty measures on the dialogue policy we perform interactive learning in a more challenging environment than the original Convlab2 template-based simulator. We add ambiguity to the simulated user utterances in the form of value variations that occur in the MultiWOZ dataset. For example, instead of the user simulator asking for a hotel for "one person", it could also say "It will be just me.". For more information see Appendix Section D.

When policies are trained for large domains, they are typically first pretrained on the corpus in a supervised manner, and then improved using reinforcement learning. We first investigate which states to use for the supervised pretraining (Section 3.3): oracle states, i.e., the dialogue state labels from the MultiWOZ corpus, or estimated belief states, e.g., those predicted by a EnD-SetSUMBT model. We then evaluate the pretrained policies with the simulated user. During the evaluation both policies use a EnD-SetSUMBT model to provide belief states. We observe that the policy pretrained using the oracle state achieves a success rate of 36.50% in the simulated environment compared to the 46.08% success rate achieved by the policy pretrained using EnD-SetSUMBT. Thus, all our following experiments use predicted belief states of respective tracking models for the pretraining stage.

For each setting of the belief tracker we have four possible belief state settings, i.e., the binary state (no uncertainty), the confidence score state, the confidence score state with additional total uncertainty features and the confidence score state with additional knowledge uncertainty features. For each setting we evaluate the policies through interaction with the user simulator, results are given in Table 2.

In interaction with the simulator, systems making use of confidence outperform the systems without any uncertainty (significance at $p < 0.05$). Moreover, the additional total and knowledge uncertainty features always outperform the systems which only use a confidence score (significance at $p < 0.05$). This indicates that additional measures of uncertainty improve the robustness of the downstream dialogue policy in a challenging environment.

It is interesting to note that the system which makes use of total uncertainty appears to outperform the system that makes use of knowledge uncertainty (significance at $p < 0.05$). We suspect that this controlled simulated environment has low data uncertainty, so the total uncertainty is overall more informative.

[4] https://gitlab.cs.uni-duesseldorf.de/general/dsml/setsumbt-public.git

4.3 Human Trial

We conduct a human trial, where we compare SetSUMBT as the baseline with EnD-SetSUMBT and EnD^2-SetSUMBT. For EnD-SetSUMBT, we consider the model that includes both confidence scores and entropy features. For EnD^2-SetSUMBT, we investigate the model that includes confidence scores and knowledge uncertainty features. For each model we have two variations: one with a binary state corresponding to the most likely state (no uncertainty variation), and one with uncertainty measures (uncertainty variation). For each variation we chose the policy whose performance on the simulated user is closest to the average performance of its respective setting, see Section 4.2.

Subjects are recruited through the Amazon Mechanical Turk platform to interact with our systems via the DialCrowd platform (Lee et al., 2018). Each assignment consists of a dialogue task and two dialogues to perform. The task comprises a set of constraints and goals, for example finding the name and phone number of a guest house in the downtown area. We encourage the subjects to use variants of labels by introducing random value variants in the tasks. The two dialogues are performed in a random order with two variations of the same model, namely no-uncertainty and uncertainty variation, as described above. After each dialogue, the subject rates the system as successful if they think they received all the information required and all constraints were met. The subjects rate each system on a 5 point Likert scale. In total we collected approximately 550 dialogues for each of 6 different systems, 3300 in total. There was a total of 380 subjects who took part in these experiments.

Table 3 shows the performance of the above policies in the human trial. We confirm that each no uncertainty system is always worse than its uncertainty counterpart (each significant at $p < 0.05$). It is important to emphasise here that in each pairing, the systems have exactly the same JGA, but their final performance can be very different in terms of success and user rating. This empirically demonstrates the limitations of JGA as a single measure for dialogue state tracking, urging the modelling of uncertainty and utilisation of calibration measures. Finally, we observe that adding additional uncertainty measures improves the policy (each significant at $p < 0.05$) and the best overall performance is achieved by the system that utilises both knowledge uncertainty and confidence scores

Belief Tracker	Belief state uncertainty	Success Rate	Rating	Turns
SetSUMBT	None	48.99	2.68	7.28
	Confidence	67.05	3.47	**6.12**
EnD-SetSUMBT	None	64.09	3.29	6.45
	Total	68.25	3.36	6.45
EnD^2-SetSUMBT	None	66.25	3.35	6.25
	Knowledge	**71.61**	**3.52**	6.31

Table 3: Performance of the systems evaluated with real users. We have 550 dialogues for each system with success rates, ratings and the number of turns averaged.

(significant at $p < 0.05$). This suggests that in human interaction there is more data uncertainty, necessitating the knowledge uncertainty to be an explicit part of the model.

It is important to note here that solely a lower average number of turns is not necessarily an indicator of the desired behaviour of a system. For example, a system which says goodbye too early may never be successful, but will have a low average number of turns.

5 Conclusion

Whilst neural dialogue state trackers may achieve state-of-the-art performance in the isolated dialogue state tracking task, the absence of uncertainty estimates may lead to less robust performance of the downstream dialogue policy. In this work we propose the use of total and knowledge uncertainties along with confidence scores to form a dialogue belief state. We moreover describe a model, SetSUMBT, that can produce such a belief state via distillation. Experiments with both simulated and real users confirm that these uncertainty metrics can lead to more robust dialogue policy models. In future, we will investigate modifying span-based dialogue state trackers to incorporate uncertainty. We will further investigate the expansion of the SetSUMBT model to include the correlation between different domain-slot pairings.

Acknowledgements

CVN, MH, NL and SF are supported by funding provided by the Alexander von Humboldt Foundation in the framework of the Sofja Kovalevskaja Award endowed by the Federal Ministry of Education and Research. CG and HCL are supported by funds from the European Research Council (ERC) provided under the Horizon 2020 research and innovation programme (Grant agreement No. STG2018 804636). Google Cloud and HHU ZIM provided computational infrastructure.

References

Arsenii Ashukha, Alexander Lyzhov, Dmitry Molchanov, and Dmitry Vetrov. 2020. Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Ultes Stefan, Ramadan Osman, and Milica Gašić. 2018. Multiwoz - A Large-Scale Multi-Domain Wizard-of-Oz Dataset for Task-Oriented Dialogue Modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Guan-Lin Chao and Ian Lane. 2019. BERT-DST: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. In *Proceedings of Interspeech 2019*, pages 1468–1472.

Zhi Chen, Lu Chen, Xiaoyuan Liu, and Kai Yu. 2020. Distributed structured actor-critic reinforcement learning for universal dialogue management. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2400–2411.

Hyunjin Choi, Judong Kim, Seongho Joe, and Youngjune Gwon. 2021. Evaluation of BERT and ALBERT sentence embedding performance on downstream NLP tasks. In *Proceedings of the 25th International Conference on Pattern Recognition (ICPR)*, pages 5482–5487.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 422–428.

Yarin Gal. 2016. *Uncertainty in Deep Learning*. Ph.D. thesis, University of Cambridge.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 3, pages 1651–1660.

Milica Gašić, Simon Keizer, Francois Mairesse, Jost Schatzmann, Blaise Thomson, Kai Yu, and Steve Young. 2008. Training and evaluation of the HIS POMDP dialogue system in noise. SIGdial '08, page 112–119, USA. Association for Computational Linguistics.

Milica Gašić and Steve Young. 2014. Gaussian processes for POMDP-based dialogue manager optimization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 22(1):28–40.

Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gašić. 2020. Trippy: A triple copy strategy for value independent neural dialog state tracking. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 35–44, 1st virtual meeting. Association for Computational Linguistics.

Matthew Henderson, Blaise Thomson, and Steve Young. 2013. Deep neural network approach for the dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, pages 467–471, Metz, France. Association for Computational Linguistics.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014. Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In *Proceedings of the 2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 360–365. IEEE.

Matthew S Henderson. 2015. *Discriminative methods for statistical spoken dialogue systems*. Ph.D. thesis, University of Cambridge.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Jiaying Hu, Yan Yang, Chencai Chen, Liang He, and Zhou Yu. 2020. SAS: Dialogue state tracking via slot attention and slot information sharing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6366–6375.

Filip Jurčíček, Blaise Thomson, and Steve Young. 2011. Natural actor and belief critic: Reinforcement algorithm for learning parameters of dialogue systems modelled as pomdps. *ACM Transactions on Speech and Language Processing (TSLP)*, 7(3):1–26.

Hwaran Lee, Seokhwan Jo, HyungJun Kim, Sangkeun Jung, and Tae-Yoon Kim. 2020. SUMBT+LaRL: End-to-end neural task-oriented dialog system with reinforcement learning. *arXiv preprint arXiv:2009.10447*.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. SUMBT: slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5478–5483.

Kyusong Lee, Tiancheng Zhao, Alan W. Black, and Maxine Eskenazi. 2018. Dialcrowd: A toolkit for easy dialog system assessment. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 245–248, Melbourne, Australia. Association for Computational Linguistics.

Esther Levin, Roberto Pieraccini, and Wieland Eckert. 1998. Using markov decision process for learning dialogue strategies. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 1, pages 201–204. IEEE.

Shiyang Li, Semih Yavuz, Kazuma Hashimoto, Jia Li, Tong Niu, Nazneen Rajani, Xifeng Yan, Yingbo Zhou, and Caiming Xiong. 2020a. CoCo: Controllable counterfactuals for evaluating dialogue state trackers. *arXiv preprint arXiv:2010.12850*.

Ziming Li, Sungjin Lee, Baolin Peng, Jinchao Li, Julia Kiseleva, Maarten de Rijke, Shahin Shayandeh, and Jianfeng Gao. 2020b. Guided dialogue policy learning without adversarial learning in the loop. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2308–2317. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Andrey Malinin. 2019. *Uncertainty Estimation in Deep Learning with application to Spoken Language Assessment*. Ph.D. thesis, University of Cambridge.

Andrey Malinin and Mark Gales. 2018. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, pages 7047–7058.

Andrey Malinin, Bruno Mlodozeniec, and Mark Gales. 2019. Ensemble distribution distillation. In *International Conference on Learning Representations*.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017a. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788, Vancouver, Canada. Association for Computational Linguistics.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017b. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua V Dillon, Balaji Lakshminarayanan, and Jasper Snoek. 2019. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*.

Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. 2017. On fairness and calibration. *Advances in Neural Information Processing Systems*, 30:5680–5689.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. Sentence meta-embeddings for unsupervised semantic textual similarity. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7027–7034, Online. Association for Computational Linguistics.

Osman Ramadan, Paweł Budzianowski, and Milica Gašić. 2018. Large-scale multi-domain belief tracking with knowledge sharing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 432–437.

Max Ryabinin, Mark Gales, and Andrey Malinin. 2021. Scaling ensemble distribution distillation to many classes with proxy targets. *arXiv preprint arXiv:2105.06987*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347.

Ryuichi Takanobu, Hanlin Zhu, and Minlie Huang. 2019. Guided dialog policy learning: Reward estimation for multi-domain task-oriented dialog. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 100–110.

Blaise Thomson, F Jurčíček, M Gašić, Simon Keizer, François Mairesse, Kai Yu, and Steve Young. 2010. Parameter learning for POMDP spoken dialogue models. In *Proceedings of the 2010 IEEE Spoken Language Technology Workshop*, pages 271–276.

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.

Carel van Niekerk, Michael Heck, Christian Geishauser, Hsien-chin Lin, Nurul Lubis, Marco Moresi, and Milica Gasic. 2020. Knowing what you know: Calibrating dialogue belief state distributions via ensembles. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3096–3102, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems*, 29:3630–3638.

Gellért Weisz, Paweł Budzianowski, Pei-Hao Su, and Milica Gašić. 2018. Sample efficient deep reinforcement learning for dialogue systems with large action spaces. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(11):2083–2097.

Jason Williams. 2012. A belief tracking challenge task for spoken dialog systems. In *NAACL-HLT Workshop on Future directions and needs in the Spoken Dialog Community: Tools and Data (SDCTD 2012)*, pages 23–24, Montréal, Canada. Association for Computational Linguistics.

Jason D Williams and Steve Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422.

Fanghua Ye, Jarana Manotumruksa, Qiang Zhang, Shenghui Li, and Emine Yilmaz. 2021. Slot self-attentive dialogue state tracking. In *Proceedings of the Web Conference 2021*, pages 1598–1608.

Steve Young, Catherine Breslin, Milica Gašić, Matthew Henderson, Dongho Kim, Martin Szummer, Blaise Thomson, Pirros Tsiakoulis, and Eli Tzirkel Hancock. 2016. Evaluation of statistical POMDP-based dialogue systems in noisy environments. In *Situated Dialog in Speech-Based Human-Computer Interaction*, pages 3–14. Springer.

Steve Young, Jost Schatzmann, Karl Weilhammer, and Hui Ye. 2007. The hidden information state approach to dialog management. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–149. IEEE.

Jianguo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wang, S Yu Philip, Richard Socher, and Caiming Xiong. 2020. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 154–167.

Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. 2020. Convlab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 142–149, Online. Association for Computational Linguistics.

A Example dialogues

Here we provide two example dialogues from our human trial, which illustrates the advantage of using uncertainty. In Figures 3 and 4 we observe that the system makes a second request to the user to clarify the intent of the user.

User: I need to book a cheap 4 star hotel.
System: Do you have a **area** preference?
User: I need the hotel to be **eastern**.
System: Is there a specific **area** of town you are interested in?
User: The **east**.

Figure 3: Extract from a dialogue with a real user illustrating the importance of uncertainty in a dialogue.

User: Find me a hotel in the city centre.
System: Do you have a price range preference?
User: **High**
System: It is in the centre. **What price range would you like?**
User: A **highly priced** hotel.
System: Everyone seems to enjoy Acorn guest house. **It is an expensive hotel**. Is there anything else I can help you with?

Figure 4: Extract from a dialogue with a real user illustrating the importance of uncertainty in a dialogue.

B Loss Functions

Consider the classification problem with input features $\boldsymbol{x}$, oracle class y and training dataset:

$$\mathcal{D}_{\text{train}} = \{(\boldsymbol{x}^{(1)}, y^{(1)}), ..., (\boldsymbol{x}^{(N)}, y^{(N)})\},$$

consisting of N labelled examples.

B.1 Label Smoothing Loss Function

The label smoothing loss is a regularised variant of standard negative log likelihood loss. Here, instead of considering a 1-hot target $\boldsymbol{y}_I$, some noise is induced in the target distribution in the form of:

$$\boldsymbol{y}_{ls} = (1 - \epsilon)\boldsymbol{y}_I + \frac{\epsilon}{K},$$

where ϵ is the smoothing parameter, $\boldsymbol{y}_{ls}$ the noisy/smoothed targets and $\boldsymbol{y}_I$ the one hot representation of the target y. The objective is to minimise the KL divergence between the predictive distribution, $\mathtt{P}(y|\boldsymbol{x}^{(i)}, \boldsymbol{\phi})$, and the smoothed target $\boldsymbol{y}_{ls}$. That is

$$\mathcal{L}_{ls}(\boldsymbol{\phi}, \mathcal{D}_{\text{train}}) = \frac{1}{N}\sum_{i=1}^{N} \mathtt{KL}\big[\boldsymbol{y}_{ls}^{(i)} || \mathtt{P}(y|\boldsymbol{x}^{(i)}, \boldsymbol{\phi})\big]$$

B.2 Distillation Loss Functions

Here we detail the loss functions used for ensemble distillation (EnD) and ensemble distribution distillation (EnD^2) in this work.

Consider an ensemble $\{\boldsymbol{\theta}^{(1)}, ..., \boldsymbol{\theta}^{(M)}\}$ consisting of M models, with predictive posterior $\mathtt{P}(y|\boldsymbol{x}^{(i)}, \mathcal{D}_{\text{train}})$.

Standard ensemble distillation (Hinton et al., 2015) is accomplished by minimising the KL-divergence between a student model with parameters $\boldsymbol{\phi}$ and the ensemble's predictive posterior:

$$\mathcal{L}_{\text{EnD}}(\boldsymbol{\phi}, \mathcal{D}_{\text{train}}) = \frac{1}{N}\sum_{i=1}^{N} \mathtt{KL}\big[\mathtt{P}(y|\boldsymbol{x}^{(i)}, \mathcal{D}_{\text{train}}) \,||\, \mathtt{P}(y|\boldsymbol{x}^{(i)}, \boldsymbol{\phi})\big]$$

Distribution distillation is accomplished using the improved loss function proposed by Ryabinin et al. (2021). Here, we first compute a *Proxy Dirichlet Target* with Dirichlet concentration parameters $\boldsymbol{\beta}$ from the ensemble:

$$\begin{aligned}
\hat{\pi}_k(\boldsymbol{x}) &= \frac{1}{M}\sum_{m=1}^{M} \mathtt{P}(y = \omega_k|\boldsymbol{x}, \boldsymbol{\theta}^{(m)}) \\
\tilde{\beta}_0(\boldsymbol{x}) &= \frac{K-1}{2\sum_{k=1}^{K} \hat{\pi}_k(\ln \hat{\pi}_k - \sum_{m=1}^{M} \frac{\ln \pi_k^{(m)}}{M})} \\
\beta_k(\boldsymbol{x}) &= \hat{\pi}_k(\boldsymbol{x}) \cdot \tilde{\beta}_0(\boldsymbol{x}) + 1,\ \beta_0 = \sum_{k=1}^{K} \beta_k.
\end{aligned} \tag{12}$$

Given this *Proxy Dirichlet Target*, distribution distillation is done by minimising the following loss:

$$\begin{aligned}
&\mathcal{L}_{\text{EnD}^2}(\boldsymbol{\phi}, \mathcal{D}_{\text{train}}) = \\
&\frac{1}{N}\sum_{i=1}^{N} \Big[-\mathbb{E}_{\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x}^{(i)}, \boldsymbol{\phi})}[\sum_{k=1}^{K} \hat{\pi}_k^{(i)} \ln \pi_k] \\
&+ \frac{1}{\beta_0^{(i)}} \mathtt{KL}[\mathtt{p}(\boldsymbol{\pi}|\boldsymbol{x}^{(i)}, \boldsymbol{\phi}) \| \mathtt{p}(\boldsymbol{\pi}|\mathbf{1})]\Big].
\end{aligned} \tag{13}$$

C SetSUMBT Implementation Details

Here we provide details regarding the SetSUMBT model configuration and the model training configuration. Table 4 provides details about the configuration of the SetSUMBT model. Tables 5 and 6 provide details regarding the training configurations for both the single model and distillation (EnD) of SetSUMBT. For all SetSUMBT models the *Set Pooler* consists of a single convolutional layer with padding followed by a mean pooling layer.

Parameter	Value
Roberta pretrained checkpoint	`roberta-base`
Hidden size	768
SUM attention heads	12
Context tracking GRU hidden size	300
Set Pooler CNN filter size	3
Dropout rate	0.3
Maximum turn length	64
Candidate description length	12

Table 4: SetSUMBT model configuration.

Parameter	Value
Learning rate (LR)	$1e-5$
LR Scheduler warmup proportion	0.1
Batch size	3
Maximum turns per dialogue	12
Epochs	100
Early stopping criteria	25 epochs
Label smoothing ϵ	0.05
α_{goal}	1.0
α_{general}	0.2
α_{request}	0.2
α_{domain}	0.2

Table 5: Single model and EnD^2 training configurations. EnD^2 does utilise the Label smoothing ϵ.

D Variations in User Simulator Output

The user simulator used in our experiments consists of a natural language understanding (NLU) module, a rule based user agent and template based natural language generation module, all provided in the ConvLab 2 environment (Zhu et al., 2020). A pre-defined set of rules simulates the user behaviour based on the predicted semantic system actions and the resulting user actions are mapped to natural language using a pre-defined set of templates. To induce variation to the user simulator utterances and thus make understanding more difficult for the system, we utilise a set of pre-defined value variations obtained from the MultiWOZ 2.1 value map (Heck et al., 2020). For example, we can map the value, `expensive`, in the user action:

```
Inform - Restaurant - Price_range
        - expensive
```

to any of the following options:

```
[high end, high class, high
scale, high price, high priced,
 higher price, fancy, upscale,
  nice, expensively, luxury ].
```

In our experiments 20% of simulated user actions contain such variations.

Parameter	Value
Learning rate (LR)	$1e-5$
LR Scheduler warmup proportion	0.1
Batch size	3
Maximum turns per dialogue	12
Epochs	100
Early stopping criteria	25 epochs
Distribution smoothing	$1e-4$
Temperature scaling base temperature	2.5
Temperature scaling annealing cycle	0.1
α_{goal}	1.0
α_{general}	0.2
α_{request}	0.2
α_{domain}	0.2

Table 6: EnD training configuration.

E System Latencies

In this section we provide the processing times per turn for our SetSUMBT model as well as the systems used in this work. These processing times are averaged across the 7372 turns in the MultiWOZ 2.1 test set. This test is performed on a Google Cloud virtual machine containing a Nvidia V100 16GB GPU, 8 n1-standard VCPU's and 30GB memory. In Table 7 we compare the latencies of a single instance of SetSUMBT against a 10 model ensemble. In Table 8 we compare the latencies of the full dialogue system setups used in this work.

Tracker	Latency (ms)
Single instance	77.7680
10 Instance ensemble	768.0256

Table 7: Turn level latency of the SetSUMBT model and ensemble.

System	Latency (ms)
No uncertainty	135.9768
Confidence scores	138.0960
Total uncertainty	147.4574
Knowledge uncertainty	152.5392

Table 8: Turn level latency of the various full dialogue systems utilised in this work.