A Confidence-based Acquisition Model for Self-supervised Active Learning and Label Correction

Carel van Niekerk, Christian Geishauser, Michael Heck, Shutong Feng, Hsien-chin Lin, Nurul Lubis, Benjamin Ruppik, Renato Vukovic, and Milica Gašić
Heinrich Heine Universität Düsseldorf, Düsseldorf, Germany
{cvanniekerk,geishaus,heckmi,fengs,linh,lubis,ruppik,revuk100,gasic}@hhu.de

Abstract

Supervised neural approaches are hindered by their dependence on large, meticulously annotated datasets, a requirement that is particularly cumbersome for sequential tasks. The quality of annotations tends to deteriorate with the transition from expert-based to crowd-sourced labeling. To address these challenges, we present **CAMEL** (**C**onfidence-based **A**cquisition **M**odel for **E**fficient self-supervised active **L**earning), a pool-based active learning framework tailored to sequential multi-output problems. CAMEL possesses two core features: (1) it requires expert annotators to label only a fraction of a chosen sequence, and (2) it facilitates self-supervision for the remainder of the sequence. By deploying a label correction mechanism, CAMEL can also be utilized for data cleaning. We evaluate CAMEL on two sequential tasks, with a special emphasis on dialogue belief tracking, a task plagued by the constraints of limited and noisy datasets. Our experiments demonstrate that CAMEL significantly outperforms the baselines in terms of efficiency. Furthermore, the data corrections suggested by our method contribute to an overall improvement in the quality of the resulting datasets.[1]

1 Introduction

Supervised training of deep neural networks requires large amounts of accurately annotated data (Russakovsky et al., 2015; Szegedy et al., 2017; Li et al., 2020b). A particularly challenging scenario arises when training for sequential multi-output tasks. In this case, the neural network is required to generate multiple predictions simultaneously, one for each output category, at every time step throughout an input sequence. Consequently, the labeling effort increases rapidly, becoming impractical as the demand for precise and consistent labeling across each time step and output category intensifies. Therefore, a heavy dependence on human-generated labels poses significant limitations on the scalability of such systems.

A prominent example of a sequential multi-output label task for which this bottleneck is evident is dialogue belief tracking. A dialogue belief tracker is one of the core components of a dialogue system, tasked with inferring the goal of the user at every turn (Young et al., 2007). Current state-of-the-art trackers are based on deep neural network models (Lin et al., 2021; van Niekerk et al., 2021; Heck et al., 2022). These models outperform traditional Bayesian network-based belief trackers (Young et al., 2010; Thomson and Young, 2010). However, neural belief trackers are greatly hindered by the lack of adequate training data. Real-world conversations, even those pertaining to a specific task-oriented domain, are extremely diverse. They encompass a broad spectrum of user objectives, natural language variations, and the overall dynamic nature of human conversation. While there are many sources for dialogue data, such as logs of call centers or virtual personal assistants, *labeled* dialogue data is scarce and several orders of magnitude smaller than, say, data for speech recognition (Panayotov et al., 2015) or translation (Bojar et al., 2017). Although zero-shot trackers do not require large amounts of labeled data, they typically underperform compared to supervised models that are trained on accurately labeled datasets (Heck et al., 2023).

One of the largest available labeled datasets for task-oriented dialogues is MultiWOZ, which is a multi-domain dialogue dataset annotated via crowdsourced annotators. The challenges in achieving consistent and precise human annotations are apparent in all versions of MultiWOZ

[1]The code is available under https://gitlab.cs.uni-duesseldorf.de/general/dsml/camell.git.

Transactions of the Association for Computational Linguistics, vol. 13, pp. 167–187, 2025. https://doi.org/10.1162/tacl_a_00734
Action Editor: Wenjie (Maggie) Li. Submission batch: 2/2024; Revision batch: 6/2024–9/2024; Published 2/2025.

(Budzianowski et al., 2018; Eric et al., 2020; Zang et al., 2020; Han et al., 2021; Ye et al., 2022). Despite manual corrections in the most recent edition, model performance has plateaued, not due to limitations in the models, but as a result of data inconsistencies (Li et al., 2020a).

Addressing the omnipresent issue of unreliable labels, as evident in the MultiWOZ dataset, is a common problem that affects the quality and reliability of supervised learning systems. In order to mitigate these issues and enhance the robustness of model training, we propose a novel methodology.

In this work, we present CAMEL, a pool-based semi-supervised active learning approach for sequential multi-output tasks. Given an underlying supervised learning model that can estimate confidence in its predictions, CAMEL substantially reduces the required labeling effort. CAMEL comprises:

- A selection component that selects a subset of time-steps and output categories to be labeled in input sequences by experts rather than whole sequences, as is normally the case.
- A self-supervision component that uses self-generated labels for the remaining time-steps and output categories within selected input sequences.
- A label validation component which examines the reliability of the human-provided labels.

We first apply CAMEL within an idealized setting for machine translation, a generative language modeling task. CAMEL achieves impressive results, matching the performance of a model trained on the full dataset while utilizing less than 60% of the expert-provided labels. Subsequently, we apply CAMEL to the dialogue belief tracking task. Notably, we achieve 95% of a tracker's full-training dataset performance using merely 16% of the expert-provided labels. Additionally, we propose an adaptation of the meta-post-hoc model approach (Shen et al., 2023), tailored for cost-efficient active learning. We demonstrate that CAMEL, utilizing uncertainty estimates from this cost-effective method, exhibits similar performance compared to using uncertainty estimates from a significantly more computationally expensive ensemble of models.

On top of this framework, we develop a method for automatically detecting and correcting inaccuracies of human labels in datasets. We illustrate that these corrections boost performance of distinct tracking models, overcoming the limitations imposed by labeling inconsistencies. Having demonstrated its efficacy in machine translation and dialogue belief tracking, our framework holds potential for broad applicability across various sequential multi-output tasks, such as object tracking, pose detection, and language modeling.

2 Related Work

2.1 Active Learning

Active learning is a machine learning framework that pinpoints scenarios in data that lack representation and interactively queries a designated annotator for labels (Cohn et al., 1996). The framework uses an acquisition function to identify the most beneficial data points for querying. Such a function estimates how performance can improve following the labeling of data. Functions of this kind often rely on various factors, such as prediction uncertainty (Houlsby et al., 2011), data space coverage (Sener and Savarese, 2018), variance reduction (Johansson et al., 2007), or topic popularity (Iovine et al., 2022).

Active learning approaches can be categorized into stream-based and pool-based (Settles, 2009). Stream-based setups are usually employed when data creation and labeling occur simultaneously. In contrast, pool-based approaches separate these steps, operating under the assumption that an unlabeled data pool is available.

Active learning has been frequently employed in tasks such as image classification (Houlsby et al., 2011; Gal et al., 2017) and machine translation (Vashistha et al., 2022; Liu et al., 2018). A noteworthy example in machine translation is the work of Hu and Neubig (2021), which enhances efficiency by applying active learning to datasets enriched with frequently used phrases. While this strategy does reduce the overall effort required for labeling, it inherently limits the scope of the annotator's work to phrases only. As a result, this method may not support the annotation of longer texts, where understanding the context and nuances of full sentences is crucial.

At the same time, active learning is less prevalent in dialogue belief tracking, with Xie

et al. (2018) being a notable exception. Their framework involves querying labels for complete sequences (dialogues) and bases selection on a single output category, neglecting any potential correlation between categories. Furthermore, this approach does not account for annotation quality problems.

One work that addresses the issue of annotation quality within an active learning framework is Su et al. (2018). In that work, stream-based active learning is deployed for the purpose of learning whether a dialogue is successful. The user-provided labels are validated using a label confidence score. This innovative learning strategy is however not directly applicable to sequential multi-output tasks, as it does not deal with the sequential nature of the problem.

2.2 Semi-Supervised Learning

Semi-Supervised Learning (SSL) makes use of both labeled and unlabeled data to improve learning efficiency and model performance. While SSL traditionally encompasses various approaches, including encoder-decoder architectures, alternative methods incorporate self-labeling or self-supervision to enhance model training with minimal human intervention.

In SSL, a ''pre-trained'' model typically undergoes an initial phase of unsupervised learning, leveraging large volumes of unlabeled data to learn representations. Subsequently, the model is fine-tuned for specific tasks using labeled data. This fine-tuning process, especially prevalent in state-of-the-art transformer-based models like RoBERTa (Liu et al., 2019), is integral to semi-supervised learning strategies, serving as an illustration of their practical utility (van Niekerk et al., 2021; Su et al., 2022; Heck et al., 2022).

Moreover, SSL can utilize self-training techniques, such as Pseudo Labeling and Noisy Student Training, where a ''teacher'' model generates pseudo labels for unlabeled data, which are then used to train a ''student'' model. In this iterative process, the student assumes the teacher role. This semi-supervised training can improve performance without necessitating extra labels.

The Pseudo-Label method proposed by Lee (2013) is a straightforward and effective SSL technique where the model's confident predictions on unlabeled data are treated as ground truth labels. This method has been widely adopted due to its simplicity and effectiveness in various domains.

Recent advances in SSL have focused on methods such as FixMatch (Sohn et al., 2020), which simplifies the semi-supervised learning pipeline by combining consistency regularisation and pseudo-labeling. FixMatch leverages weakly augmented data to predict pseudo labels, and strongly augmented data to enforce consistency.

Additionally, Xie et al. (2020) propose the Noisy Student method, which extends the teacher-student framework by adding noise to the student model, thereby improving its robustness and performance. Further, Kumar et al. (2020) explore the concept of gradual domain adaptation through self-training, where a model is iteratively trained on data that gradually shifts from the source to the target domain. This approach has been shown to effectively handle large distribution shifts by leveraging intermediate domains to improve generalisation.

In summary, the incorporation of self-supervision and iterative training frameworks in SSL has proven to be highly effective, driving advancements in model performance with minimal labeled data. These methods not only enhance the learning process but also reduce the reliance on extensive labeled datasets, making SSL a crucial area of research in modern machine learning.

2.3 Label Validation

The process of manually correcting labels is very tedious and expensive. As a result, many works focus on learning from imperfect labels, using loss functions and/or model architectures adapted for label noise (Reed et al., 2015; Xiao et al., 2015; Sukhbaatar et al., 2015). Still, these methods have been unable to match the performance of models trained on datasets that include manually corrected labels. However, the alternative of automated label validation or correction is often overlooked by such works. It has been shown that learning from automatically corrected labels, e.g., based on confidence scores, performs better than learning from noisy labels alone (Liu et al., 2017; Jiao et al., 2019). The major drawback of these approaches is that they frequently rely on over-confident predictions of neural network models to correct labels, which can further bias the model.

3 CAMEL: Confidence-based Acquisition Model for Efficient Self-supervised Active Learning

In this section, we introduce our pool-based active learning approach, named *CAMEL*, to address sequential multi-output classification problems. Let us consider a classification problem with input features $\boldsymbol{x}$, and output $\boldsymbol{y}$. According to Read et al. (2015), such a problem can be cast as a *multi-output* classification problem if the output consists of multiple label categories that need to be predicted simultaneously. Specifically, for a problem with M categories, the output is represented as $\boldsymbol{y} = \langle y^1, y^2, \ldots, y^M \rangle$, where each $y^m, m \in [1, M]$ can be binary or multivariate. Furthermore, this problem is characterized as a *sequential* classification problem if the output is dependent on a sequence of prior inputs. For a sequence with T time-steps, the input-output pairs can be represented as $\langle (\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2), \ldots, (\boldsymbol{x}_T, \boldsymbol{y}_T) \rangle$, where $\boldsymbol{y}_t = \langle y_t^1, y_t^2, \ldots, y_t^M \rangle$ represents the output labels at time step $t \in [1, T]$.

In a conventional setting, for an unlabeled data sequence $\boldsymbol{X}_i = \langle \boldsymbol{x}_1, \ldots, \boldsymbol{x}_{T_i} \rangle$, an annotator would typically be required to provide labels, y_t^m, for each label category m at every time step t, which is considerably expensive.

3.1 Requirements

CAMEL, as a confidence-based active learning framework, utilizes confidence estimates to determine data points to be queried for labeling. The framework relies on the model's ability to gauge the certainty of each prediction. Specifically, for every time-step t in a sequence, for each category m in a multi-output setting, and for each possible value $v \in \mathcal{V}^m$ that m can take, the model calculates the predictive probability, $\pi_t^m(v) = \mathrm{p}\left(y_t^m = v\right)$. These probabilities, collected into a distribution $\boldsymbol{\pi}_t^m = [\pi_t^m(v)]_{\forall v \in \mathcal{V}^m}$, form the predictive distribution that CAMEL uses for active learning decisions.

The calibration of these confidence estimates is also critical. Calibration refers to the alignment between the model's estimated confidence and the empirical likelihood of its predictions (Desai and Durrett, 2020). Should the model's confidence estimates be poorly calibrated, it may select instances that are not informative, resulting in an inefficient allocation of the annotation budget and potentially suboptimal performance.

3.2 Active Learning Approach

The approach we propose starts with an initial learning model, which is trained using a small labeled *seed* dataset and iteratively progresses through four stages: data selection, labeling, label validation, and semi-supervised learning. These iterations continue until either a pre-defined performance threshold is achieved or the dataset is fully labeled. The schematic representation of this approach is illustrated in Figure 1.

Stage 1: Data Selection In each cycle, we select a subset of N_{sel} sequences from the unlabeled pool of size N_{unlb}. Selection is based on the model's prediction confidence, p_t^m (which will be specified in Equation 1). Instances in which the model displays low confidence (confidence below a threshold α_{sel}) are selected. More precisely, an input sequence is selected if the model shows high uncertainty for at least one time-step t and label category m instance y_t^m. The α_{sel} threshold is set such that N_{sel} sequences are selected for labeling.

Stage 2: Labeling In the input sequences selected in Stage 1, the learning model self-labels the time-steps and categories, $\hat{v}_t^m = \arg\max_{v \in \mathcal{V}^m}(\pi_t^m(v))$, where its confidence is above the threshold α_{sel}. Concurrently, expert annotators are responsible for labeling the remaining time-steps and categories. These labels are denoted by $\tilde{v}_t^m$.

Stage 3: Label Validation This is an optional step, and the variant of CAMEL that contains this stage we call Confidence-based Acquisition Model for Efficient Self-supervised Active Learning with Label Validation (CAMELL). We can consider the labels, $\tilde{v}_t^m$, with label confidence, $\tilde{\mathrm{p}}_t^m$, below a threshold α_{val} to be potentially incorrect. This label confidence is not assigned by the annotators themselves but is computed by the learning model. To safeguard the model from being trained with these potentially erroneous labels, we purposely exclude them (i.e., these labels are masked in the dataset). The α_{val} threshold can be set using a development set.

Stage 4: Semi-supervised Learning At each iteration of the active learning approach, the expert provided labels that passed validation (Stage 3) and the self-determined labels from Stage 2 are added to the labeled pool, resulting in $N_{\text{lab}} + N_{\text{sel}}$ data sequences. Based on these, the learning model is retrained.

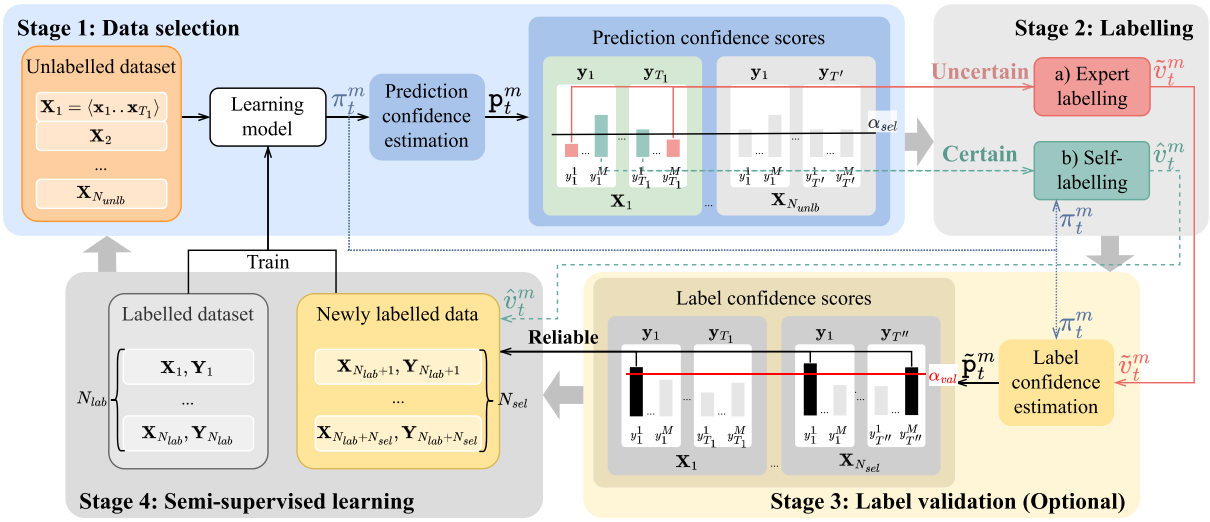


Figure 1: CAMEL comprises four stages. Stage 1 involves data selection, choosing instances for labeling where the model shows uncertainty (confidence below the α_{sel} threshold), as indicated by pink arrows. In Stage 2, annotators label the selected instances while the model self-labels the remaining ones (dashed green arrows). Stage 3 (optional) validates labels using a label confidence estimate, incorporating only labels exceeding the α_{val} threshold and the self-labeled data into the dataset (black arrows). Finally, Stage 4 involves retraining the model for the next cycle.

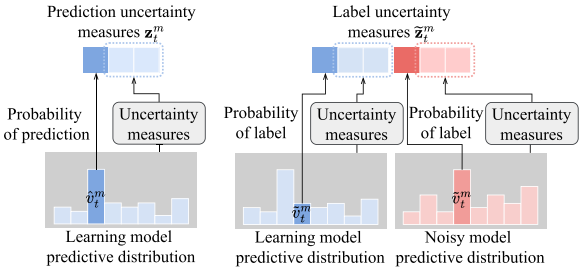
3.3 Confidence Estimation

To accurately estimate the prediction confidence required in Stage 1 as well as the label confidence in Stage 3, we propose a confidence estimation model for each stage. These models are designed to encapsulate the learning model's confidence by considering both its *total* and *knowledge-based* uncertainties. *Total* uncertainty captures all uncertainty in the model's prediction, irrespective of the source. Conversely, *knowledge* uncertainty in a model originates from its incomplete understanding, which occurs due to a lack of relevant data during training, or the inherent complexity of the problem (Gal, 2016).

Both the prediction and label confidence estimation models share the same objective: to estimate the probability that the value v_t^m for a specific label category m at time-step t is correct. To provide the training data for these models, we assume that the labels in the labeled pool are correct, as they have already been validated. Furthermore, we retrain these models whenever more data is labeled.

Both models share the same general structure:

$$\begin{aligned} \boldsymbol{h}_t^m &= \texttt{Enc}_{\texttt{Intra-Cat}}(\boldsymbol{z}_t^m) \\ \boldsymbol{h}_t &= \texttt{Enc}_{\texttt{Inter-Cat}}([\boldsymbol{z}_t^j]_{j=1}^M) \\ \mathrm{p}_t^m &= \texttt{Conf}(\boldsymbol{h}_t^m, \boldsymbol{h}_t), \end{aligned} \tag{1}$$



(a) Prediction (b) Label

Figure 2: Category-specific uncertainty measures: (a) displays prediction uncertainty, including prediction probability and total and knowledge uncertainty; (b) depicts label uncertainty, including label probability and total and knowledge uncertainty from both learning and noisy models.

where $\boldsymbol{z}_t^m = [\pi_t^m(v_t^m), \mathcal{T}(\boldsymbol{\pi}_t^m), \mathcal{K}(\boldsymbol{\pi}_t^m)]$ is a set of uncertainty measures for category m. As illustrated in Figure 2, these measures consist of the predictive probability specific to $\pi_t^m(v_t^m)$, along with measures of *total* uncertainty, $\mathcal{T}(\boldsymbol{\pi}_t^m)$, and *knowledge* uncertainty, $\mathcal{K}(\boldsymbol{\pi}_t^m)$, associated with the predictive distribution $\boldsymbol{\pi}_t^m$ (See Sections 4.4.1 and 4.5.1 for concrete implementations). The *intra-category encoder* is tasked with extracting important category specific features, $\boldsymbol{h}_t^m$, from these uncertainties. Important features across categories, $\boldsymbol{h}_t$, are extracted by the *inter-category*

encoder.[2] The *inter-category* encoder allows the model to take advantage of any correlations between categories, which was not done by Xie et al. (2018). Both the *inter-* and *intra-category* encoders consist of linear fully connected layers. The *confidence estimation* component generates a confidence score, p_t^m, reflecting the accuracy of a given category's value. This component is composed of a linear feature transformation layer, followed by a prediction layer with a Sigmoid activation function.

The design choices for the confidence estimation models were motivated by a desire to capture both *intra-* and *inter-category* uncertainty for reliable confidence estimation. We observed that excluding *inter-category* features degraded performance, emphasizing the importance of incorporating them.

3.3.1 Prediction Confidence Estimation

The objective of the prediction confidence estimation model is to assess whether the value predicted by the learning model, $\hat{v}_t^m$, is the ''true'' value, based on the prediction confidence score p_t^m. This model, also known as the confidence-based acquisition model, is used as the selection criterion in Stage 1.

3.3.2 Label Confidence Estimation

The objective of the label confidence estimation model is to determine whether an annotator's label, $\tilde{v}_t^m$, is the ''true'' value, with this decision being based on the label confidence score $\tilde{\mathrm{p}}_t^m$. In Su et al. (2018), the confidence score of the learning model is directly used for both purposes. We believe this is a suboptimal strategy, because the model has not been exposed to instances of ''incorrect'' labels. To address this, we generate a *noisy* dataset featuring ''incorrect'' labels for training purposes.

Further, we extend $\tilde{\boldsymbol{z}}_t^m$ to include uncertainty measures drawn from both a *noisy* model, trained on the corresponding *noisy* dataset, and the original learning model (as depicted in Figure 2b). Given that the *noisy* model is conditioned to accept the ''incorrect'' labels as correct, the discrepancy in uncertainty between the *noisy* model and the learning model enhances the label confidence estimator's ability to identify potentially incorrect labels.

Noisy Dataset The creation of a noisy dataset can be approached in two ways. One method is to randomly replace a portion of labels. However, this approach may not yield a realistic noisy dataset, considering human errors are rarely random. A second approach, particularly when the learning model is an ensemble, as is often the case for uncertainty-endowed deep learning models (Gal and Ghahramani, 2016; Ashukha et al., 2020), is to leverage individual ensemble members to supply noisy labels (see Section 4.5.1 for details related to an ensemble free approach). This method may be more effective, given the individual members' typical lower accuracy compared to the ensemble as a whole.

In our proposed approach, we initially select α_{noise} percent of the sequences from the training data at random. For each category m, we choose a random ensemble member to generate noisy labels. This ensemble member creates labels at each time step t by sampling from its predictive probability distribution. To avoid generating labels from the *clean* dataset, the probabilities of these are set to zero prior to sampling. The noisy dataset is regenerated after each update of the learning model using the updated ensemble members, enhancing diversity of noisy labels.

3.4 Label Correction

We propose a label correction method that utilizes the model that solves the task at hand, referred to as the learning model, the label confidence estimation model (Section 3.3.2), and the prediction confidence estimation model (Section 3.3.1). In order to correct a noisy dataset, this method involves three steps: (1) detecting potentially erroneous labels, (2) determining which of these labels can be accurately corrected by the learning model, and (3) substituting the incorrect labels with the learning model's predictions. Detecting potentially erroneous labels requires utilizing the label confidence estimation model and setting the hyperparameter α_{val}, such that all labels $\tilde{v}_t^m$ with confidence below this threshold are considered potentially incorrect. Then the prediction confidence model is utilized to estimate the learning model's confidence of detected erroneous labels $\tilde{v}_t^m$. If this confidence is greater than the one assigned by the

[2] During label confidence estimation, for categories not selected for labeling, self-labels are used to complete the *inter-category* features.

label confidence estimation model, the labels are substituted with the learning model's predictions.

3.5 Efficient Confidence Estimation with Post-hoc Uncertainty Learning

To obtain reliable estimates of the knowledge and total uncertainties required in Section 3.3, an ensemble-based approach is typically employed; however, this method is computationally expensive (Gal, 2016). This challenge is amplified in active learning scenarios, where the model is frequently updated. Shen et al. (2023) propose an uncertainty estimation technique in which uncertainties are generated by a post-hoc Dirichlet meta-model, offering greater computational efficiency than an ensemble of models. This method enables the model to distinguish between knowledge and data uncertainty, without needing several instances of the learning model. The post-hoc Dirichlet meta-model involves a two-stage training process. In the initial stage, a model with the same architecture as the learning model is trained to create a base model. In the second stage, meta-features are employed to estimate the uncertainties of the base model. These meta-features, derived from various intermediate layers of the base model, capture distinct levels of feature representation, from low- to high-level representations. Utilizing the diversity in these representations allows for more nuanced uncertainty quantification (Shen et al., 2023). To capture the uncertainty of the base model, we utilize a meta-model. This meta-model takes as input the intermediate features from the base model and outputs the parameters of a Dirichlet distribution. This Dirichlet distribution over the probability simplex, in turn, describes the uncertainty present in the prediction.

More rigorously, given a base neural network model that solves the task at hand, the set of L features $\boldsymbol{F} = \{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_L\}$ is extracted from different layers of this model for a given input, where L refers to the number of layers of the base model. These intermediate features can include embeddings from various layers within a neural network, such as the transformer layers in a transformer model. Meta-features are computed via small meta-feature extraction layers, g_l.

In our case, these are fully connected layers with a ReLU activation function that map the intermediate features to meta-features of dimension $d_{\texttt{meta}}$, $\boldsymbol{m}_l = g_l(\boldsymbol{f}_l)$ for $l = 1, \ldots, L$. These meta-features are then combined and mapped to the required prediction dimension through another fully connected layer with ReLU activation.

3.5.1 Learning Objective

The post-hoc meta-model is trained using Bayesian matching loss (Joo et al., 2020) with the same training dataset as the base model.

The loss for the meta-model, denoted as $\mathcal{L}_{\texttt{meta}}$, is defined as:

$$\begin{aligned}
&\mathcal{L}_{\texttt{meta}}\left(\boldsymbol{\theta}^{(\texttt{meta})}; \mathcal{D}\right) \\
&= \mathbb{E}_{\mathrm{p}(\boldsymbol{x},y|\mathcal{D})}\left[\mathbb{E}_{\mathrm{p}\left(\boldsymbol{\pi}|\boldsymbol{x},\boldsymbol{\theta}^{(\texttt{meta})}\right)}\left[-\log \mathrm{p}\left(y \mid \boldsymbol{\pi}\right)\right]\right] \\
&+\lambda \mathbb{E}_{\mathrm{p}(\boldsymbol{x},y|\mathcal{D})}\left[D_{\mathrm{KL}}\left[\mathrm{p}\left(\boldsymbol{\pi}|\boldsymbol{x},\boldsymbol{\theta}^{(\texttt{meta})}\right) \middle\| \mathrm{p}\left(\boldsymbol{\pi}|\boldsymbol{\beta}\right)\right]\right].
\end{aligned}$$

In this expression, the first term represents the expected negative log-likelihood. The second term, involving the Kullback-Leibler (KL) divergence, quantifies the deviation of the model's predictive distribution from a Dirichlet prior. This prior, $\mathrm{p}\left(\boldsymbol{\pi} \mid \boldsymbol{\beta}\right)$, represents our belief about the uncertainty before observing the data.

We can show that an optimal state for this model is reached when the output, $\widehat{\boldsymbol{\alpha}}$, equals the sum of the prior concentration parameters and the scaled one-hot encoded label, $\widehat{\boldsymbol{\alpha}} = \boldsymbol{\beta} + \frac{1}{\lambda}\boldsymbol{y}$. This mechanism enables the model to adjust its uncertainty by integrating both prior knowledge and the evidence gathered from observed data. However, the reliance on constant prior concentration parameters, $\boldsymbol{\beta}$, introduces a limitation. Specifically, it encourages the model to generate similar uncertainty estimates across all inputs, irrespective of their complexity. This, however, leads to a model that is under-confident for inputs it can correctly predict and over-confident for inputs it cannot. To address this problem, we introduce a distillation approach called *Dynamic Priors* within the Bayesian matching loss framework. Dynamic Priors adapt at each active learning step by leveraging previous model versions, thereby mitigating the constant prior problem.

3.5.2 Dynamic Priors

Dynamic priors leverage the active learning setting in which we operate. This setting allows the model to access previous versions of the learning model, which can then be used as priors. The underlying hypothesis is that replacing the constant prior, as described in Section 3.5.1, with a dynamic prior—one that evolves at each active learning

step—addresses the homogeneity issue discussed above.

More concretely, the prior is predicted from the Dirichlet distributions from the previous model version. If no previous version is available, such as at the beginning of the active learning process, a small ensemble of models, $\left\{\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots, \boldsymbol{\theta}^{(E)}\right\}$, trained on a small seed-set from the active learning initialization phase, is used to obtain the initial prior. It is important to emphasize that only the initial prior is obtained using a small ensemble. In all subsequent updates to the model, the predicted Dirichlet distribution from a single model instance is used as the prior. By parameterizing the Dirichlet prior, $\mathtt{p}\left(\boldsymbol{\pi} \mid \boldsymbol{\beta}\right)$, with the previous model's outputs, our approach dynamically adjusts the prior concentration parameters. This adjustment not only mitigates the issue of constant priors but also increases the model's ability to produce more accurate uncertainty estimates.

In order to represent the knowledge of the ensemble using a Dirichlet distribution, the ensemble's aggregate predictive distribution, $\widetilde{\boldsymbol{\pi}}(\boldsymbol{x}) = \frac{1}{E}\sum_{e=1}^{E}\boldsymbol{\pi}^{(e)}(\boldsymbol{x})$, and the individual distributions, $\boldsymbol{\pi}^{(e)}(\boldsymbol{x}) = \mathtt{p}\left(y \mid \boldsymbol{x}, \boldsymbol{\theta}^{(e)}\right)$, are utilized to compute the prior concentration parameters, $\boldsymbol{\beta}(\boldsymbol{x})$. Specifically, following Ryabinin et al. (2021), we use Stirling's approximation. $\boldsymbol{\beta}(\boldsymbol{x})$ is defined as $\beta_0(\boldsymbol{x}) \cdot \widetilde{\boldsymbol{\pi}}(\boldsymbol{x})$, where $\beta_0(\boldsymbol{x})$ is defined as:

$$\beta_0(\boldsymbol{x}) = \frac{K-1}{2\sum_{k=1}^{K}\tilde{\pi}_k(\boldsymbol{x}) \cdot d_k(\boldsymbol{x})}, \text{ with}$$
$$d_k(\boldsymbol{x}) = \log\tilde{\pi}_k(\boldsymbol{x}) - \frac{1}{E}\sum_{e=1}^{E}\log\pi_k^{(e)}(\boldsymbol{x}).$$

In the context of active learning, this approach allows our meta-model to incorporate new labels from annotators while retaining the rich uncertainty estimates derived from the ensemble. As the learning progresses, the priors are continually updated with predictions from the latest model, ensuring that the uncertainty estimates remain current.

Given the Dirichlet distribution $\mathtt{Dir}\left(\boldsymbol{\alpha}\right)$ produced by the post-hoc meta-model, the total and knowledge uncertainties can be approximated as follows:

$$\mathcal{T}\left(\boldsymbol{\pi}\right) = \mathcal{H}\left[\frac{\boldsymbol{\alpha}}{\alpha_0}\right],$$
$$\mathcal{K}\left(\boldsymbol{\pi}\right) = \mathcal{T}\left(\boldsymbol{\pi}\right) + \sum_{k=1}^{K}\frac{\alpha_k}{\alpha_0}\left[\psi\left(\alpha_k^*\right) - \psi\left(\alpha_0^*\right)\right].$$

Here $\boldsymbol{\pi} \sim \mathtt{Dir}\left(\boldsymbol{\alpha}\right)$, $\psi(\cdot)$ represents the digamma function, $\alpha_i^* = \alpha_i + 1$, and $\mathcal{H}[\cdot]$ denotes the entropy of a distribution. Further, to generate noisy data, predictive distributions are sampled from the Dirichlet distribution to simulate different ensemble members.

Finally, it is important to emphasize the computational efficiency of this approach. During training, only the parameters of the meta-model, which typically constitute less than 5% of the base model's size, are updated. Additionally, in the inference phase, the meta-model incurs an additional computational cost of approximately 15–20% of the total inference cost, resulting in an overall computationally efficient approach to uncertainty estimation.

4 Experiments

4.1 Baselines

Random Selection randomly selects sequences to be annotated. Random selection is often used as a baseline for active learning approaches, as it allows us to observe the impact of purely adding more labeled data to our labeled pool without strategically selecting sequences to be labeled. Its advantage is that it maintains the full data distribution with every selection, thus not creating a bias (Dasgupta and Hsu, 2008).

Bayesian Active Learning by Disagreement (BALD) is an uncertainty-based active learning method which employs knowledge uncertainty as the primary metric for selection (Houlsby et al., 2011). This technique has established itself as a strong baseline in various applications. For instance, in image classification tasks (Gal et al., 2017) and named entity recognition (Shen et al., 2017), BALD has shown notable performance. Its performance is further enhanced when used in conjunction with ensemble models (Beluch et al., 2018). Given its widespread adoption and proven efficacy, we see BALD as an ideal baseline.

In our study, we examined two criteria for making the selection decision: one based on the cumulative uncertainty across all time-steps and label categories, and another based on the average uncertainty across categories and time. Upon evaluation, we observed that the latter criterion yielded superior results, and therefore, adopted it as our baseline, which we refer to as **BALD**.

We further present an enhanced version of BALD which consists of stages 1, 2, and 4 of our approach as outlined in Section 3.2, utilizing knowledge uncertainty as the *prediction confidence estimate*. We call this BALD with self-supervision, **BALD+SS**.[3]

4.2 Variants of CAMEL

We introduce the following variants to understand the individual and collective contributions of our proposed framework's components.

CAML **C**onfidence-based **A**cquisition **M**odel for active **L**earning, represents the foundational layer of our framework, incorporating stages 1, 2a, and 4 described in Section 3.2. Crucially, it excludes the self-labeling process (stage 2b), in stage 2, thus relying solely on labels from the annotators. This variant serves as a baseline to evaluate the efficacy of our confidence estimation model in an active learning context, without the influence of self-supervision. For brevity, we report the CAML results for the translation experiments only (similar trends were observed in the dialogue belief tracking task).

CAMEL **C**onfidence-based **A**cquisition **M**odel for **E**fficient self-supervised active **L**earning is the complete approach, which also includes the self-supervision component. This variant assesses the value added by self-supervision to the framework, while retaining stages 1, 2, and 4.

CAMELL **C**onfidence-based **A**cquisition **M**odel for **E**fficient self-supervised active **L**earning with **L**abel validation is an extended variation of our approach that includes a label validation component, denoted as Stage 3 in Section 3.2.

4.3 Variants of Label Correction

Live Label Correction involves simultaneous labeling, validation, and correction of data. A variant of CAMELL is employed, in which the label is corrected at the validation stage using the prediction of the learning model.

On-line Label Correction is a method that labels and validates data simultaneously, with the objective of minimising human effort in providing labels while concurrently validating them. CAMELL can be employed to flag the data points requiring correction, as well as to apply corrections to the flagged labels using the final model after active learning has been performed.

Offline Label Correction is a technique used to correct an already labeled corpus, with the objective of identifying potentially incorrect labels and providing alternatives. To achieve this, individually trained components of CAMELL can be utilized, specifically the prediction confidence model (Section 3.3.1) and the label confidence model (Section 3.3.2). The process consists of the following steps:

1. Train learning model on labeled corpus.
2. Generate noisy dataset using this model, leveraging ensemble members from Step 1. If computational constraints prevent the use of an ensemble, a noisy dataset can be generated from a single model using the strategy described in Section 3.5.
3. Train learning model on noisy dataset.
4. Train prediction and label confidence models.
5. Perform label correction.

Semi-offline Label Correction is a method in which data is collected with the objective of minimising human effort in providing labels, with validation occurring subsequently. For this purpose, CAMEL can be utilized alongside a separately trained label confidence model (Steps 2 and 3 from above), followed by Step 5.

4.4 Generative Language Modeling Task

For the generative language modeling task, we explore the application of our CAMEL framework to the task of Neural Machine Translation (NMT). NMT focuses on converting sequences of text from a source language to a target language. Our approach involves iterative annotation methods similar to those used in automatic speech recognition (Sperber et al., 2016), which incrementally increase model precision.

Specifically, in our experiment, an annotator corrects individual words within a translation, thereby progressively enhancing the quality of the subsequent output generated by the model. Conventional annotation methods typically involve

[3]Note that we are not able to combine BALD with label validation as knowledge uncertainty does not provide candidate level confidence scores.

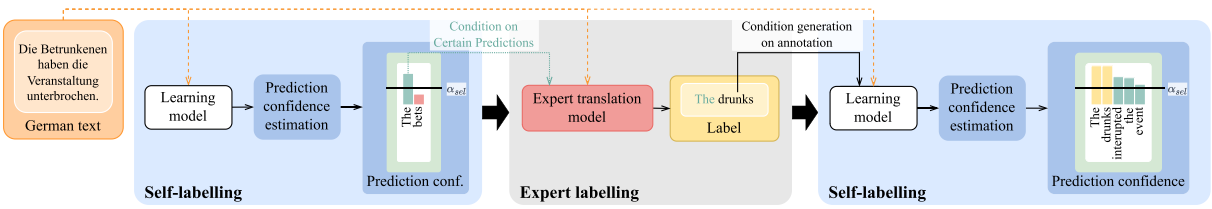


Figure 3: The model-based annotation process for semi-supervised annotation for NMT. The learning model initiates the translation with the word ‘‘The’’, then confidence for the next token generation is below the threshold. The expert annotation model is prompted and provides the next word, ‘‘drunks’’. The learning model resumes and successfully generates the remainder of the translation: ‘‘interrupted the event’’.

providing fully corrected translations or quality ratings. While this iterative process diverges from conventional methods of machine translation annotation, it allows us to effectively demonstrate the self-supervision mechanism within our framework.

4.4.1 Implementation Details

We apply CAMEL to the task of machine translation, specifically using the T5 encoder-decoder transformer model (`t5-small`) (Raffel et al., 2020). We utilize an ensemble of 10 models in order to produce a well-calibrated predictive distribution, which requires 2500 GPU hours to fully train. Approximately 40% of this time is for training the ensemble, 50% for the annotation process, and 10% for training the confidence estimator.

The ensemble model produces two types of uncertainty within the translation process. The first, termed total uncertainty, is measured by the entropy across the ensemble's predictive distribution. The second, knowledge uncertainty, is measured by the mutual information shared between the predictive distribution and the individual ensemble models. These uncertainties are crucial for evaluating the reliability of translations. The mathematical formulations for calculating total ($\mathcal{T}$) and knowledge ($\mathcal{K}$) uncertainties are as follows:

$$\mathcal{T}(\boldsymbol{\pi}) = \mathcal{H}\left[\frac{1}{E}\sum_{j=1}^{E}\boldsymbol{\pi}^{(j)}\right],$$

$$\mathcal{K}(\boldsymbol{\pi}) = \frac{1}{E}\sum_{e=1}^{E} D_{\mathrm{KL}}\left[\boldsymbol{\pi}^{(e)}\middle\|\frac{1}{E}\sum_{j=1}^{E}\boldsymbol{\pi}^{(j)}\right],$$

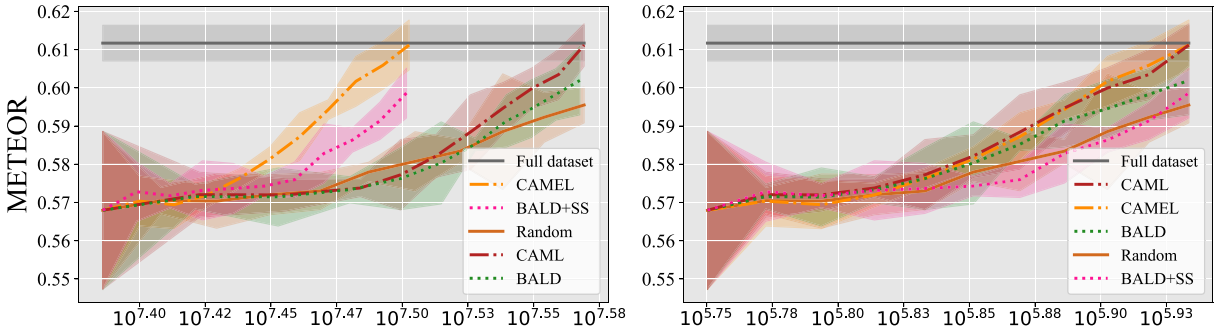
where $\boldsymbol{\pi}^{(e)}$ represents the predictive distribution from the e^{th} ensemble member.

The WMT17 DE-EN dataset, which consists of German to English translations (Bojar et al., 2017), is used for training, and METEOR (Banerjee and Lavie, 2005), BLEU (Papineni et al., 2002), and COMET (Rei et al., 2020) serve as evaluation metrics.

As machine translation does not entail a multi-output task, we employed a simplified version of the confidence estimation model, introduced in Section 3.3, consisting of only the intra-category encoder. The latent dimension of the encoder and feature transformation layer is 16. The parameters are optimised using the standard binary negative log likelihood loss (Cox, 1958).

It is crucial to address the inherent challenges in sequential machine translation labeling: (1) future sentence structure and labels can change depending on the current label, and (2) for any word position there exist multiple valid candidate words. This complexity necessitates the use of a dynamic annotation approach, as static dataset labels are insufficient for new data labeling. To avoid high translation annotation costs, we propose a practical approach: using an *expert* translation model, specifically the MBART-50 multilingual model (Tang et al., 2020), to simulate a human annotator.

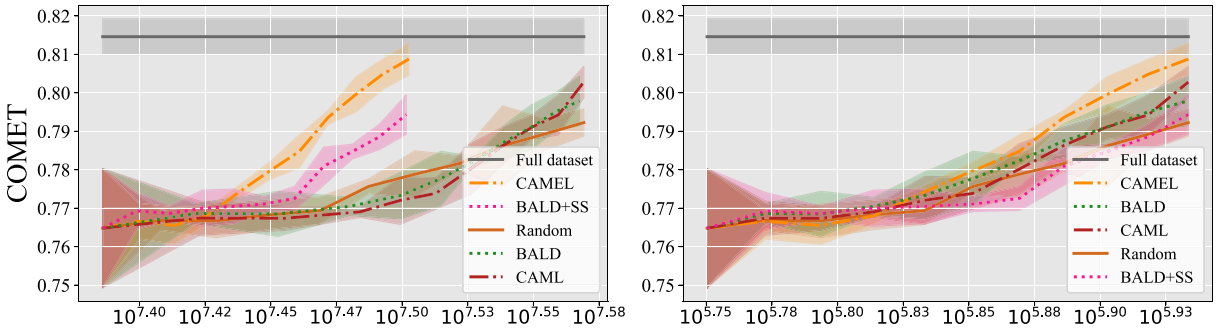
Our approach, depicted in Figure 3, is a multi-stage procedure. Initially, the learning model produces a translation for a selected source language sentence. As it generates the translation, it simultaneously estimates its confidence for the subsequent token. Should this confidence fall below a set threshold α_{sel}, the *expert* translation model steps in to supply the next word in the translation. After the label is provided, the learning model resumes the translation generation. For any future token whose confidence drops below the threshold, the *expert* translation model re-engages. This process continues until a complete translation for the source sentence is realised. The uncertainty



(a) Number of word-level labels

(b) Number of complete translations

Figure 4: METEOR score of the T5 translation model using different active learning approaches on the WMT17 DE-EN test set, as a function of (a) the number of word-level labels and (b) the number of complete translations, with 95% confidence interval.



(a) Number of word-level labels

(b) Number of complete translations

Figure 5: COMET score of the T5 translation model using different active learning approaches on the WMT17 DE-EN test set, as a function of (a) the number of word-level labels and (b) the number of complete translations, with 95% confidence interval.

threshold, α_{sel}, is strategically chosen to yield a maximum of N_{ann} word labels.

4.4.2 Results

We evaluated the performance of our proposed CAMEL framework and baseline models using METEOR (Banerjee and Lavie, 2005), BLEU (Papineni et al., 2002), and COMET (Rei et al., 2020) scores. While traditional metrics such as METEOR and BLEU highlight similar trends (with BLEU scores included in Appendix A, Figure 8), COMET, a neural evaluation metric, provides a more comprehensive understanding of the translation quality beyond traditional metrics. We establish that our proposed CAMEL framework, enhanced with self-supervision, is significantly more efficient requesting word-level labels than baseline models like BALD, BALD+SS, and random selection. This efficiency is evident in Figures 4a and 5a, which showcases CAMEL's need for fewer word-level labels to achieve similar performance. Although our primary focus is on the number of word-level labels queried, it is crucial to note that labeling overhead is also accounted for. We measure this overhead by the effort required to read and understand the source language tokens, which we consider a sufficient indicator.

A notable point to observe in Figure 4b is that the introduction of self-supervision to CAMEL does not significantly influence its performance in terms of the number of complete translations required, as evident by the comparison between CAML (CAMEL without the self-supervised labeling component) and CAMEL. This implies that self-supervision within CAMEL is applied predominantly when the model's predictions can be considered reliable. In contrast, we observe that BALD+SS, despite its label efficiency shown in Figure 4a, performs poorly in terms of the number of complete translations required, as demonstrated

Confidence Estimator	Dataset	ECE (%) ↓
CE-T5 + CAML	WMT17 DE-EN	**26.74**∗
CE-T5 + BALD	WMT17 DE-EN	47.21
CE-SetSUMBT + CAML	MultiWOZ 2.1	**9.65**∗
CE-SetSUMBT + BALD	MultiWOZ 2.1	17.21

Table 1: Comparison of the expected calibration error (ECE) of confidence estimation approaches. ∗ indicates significant difference on 95% confidence interval.

in Figure 4b. This drop in performance may be attributed to BALD+SS's tendency to incorrectly self-label complex examples. This trend is further evidenced by CAML's lower expected calibration error (ECE), reported in Table 1. The COMET results, presented in Figure 5, further attest to CAMEL's superiority. CAMEL not only excels in reducing the number of word-level labels but also outperforms other models in the number of complete translations required. The non-overlapping confidence intervals in the results indicates that the improvements of CAMEL over other methods are statistically significant.

Regardless of the methodology used, all models require roughly the same number of complete translations, as shown in Figures 4b and 5b. This supports the widely accepted notion that exposure to large datasets is vital for training robust natural language processing (NLP) models.

Encouraged by these results, we adapt CAMEL to address the dialogue belief tracking problem, a task plagued by errors in the labels of available datasets.

4.5 Dialogue Belief Tracking Task

In task-oriented dialogue, the dialogue ontology $\mathcal{O}$ contains a set of M `domain-slot` pairs $\{s^1, s^2, \ldots, s^M\}$ and a set of plausible values $\mathcal{V}_{s^m}$ for each s^m. The goal of the dialogue belief tracker is to infer the user's preference for each s^m by predicting a probability distribution over the plausible values. Notably, each set of plausible values, $\mathcal{V}_{s^m}$, includes the `not_mentioned` value, indicating that a specific `domain-slot` pair is not part of the user's goal. This allows for computing the model's confidence for slots not present in the user's preference.

To train a belief tracking model, we require the dialogue state, which includes the `value` label for each `domain-slot`, in every dialogue turn. The dialogue state at turn t in dialogue i is represented as $\mathcal{B}_{i,t} = \{(s^m, v_{i,t}^{s^m})\}_{s^m \in \mathcal{O}}$, where $v_{i,t}^{s^m}$ denotes the `value` for the `domain-slot` pair s^m at turn t in dialogue i. Consequently, we obtain a dataset $\mathcal{D} = \{(\mathbf{u}_{i,1:t}^{\text{usr}}, \mathbf{u}_{i,1:t-1}^{\text{sys}}, \mathcal{B}_{i,t})_{t=1}^{T_i}\}_{i=1}^{N}$, consisting of N dialogues, each comprising T_i turns, where user and system utterances at turn t in dialogue i are denoted as $\mathbf{u}_{i,t}^{\text{usr}}$ and $\mathbf{u}_{i,t}^{\text{sys}}$, respectively.

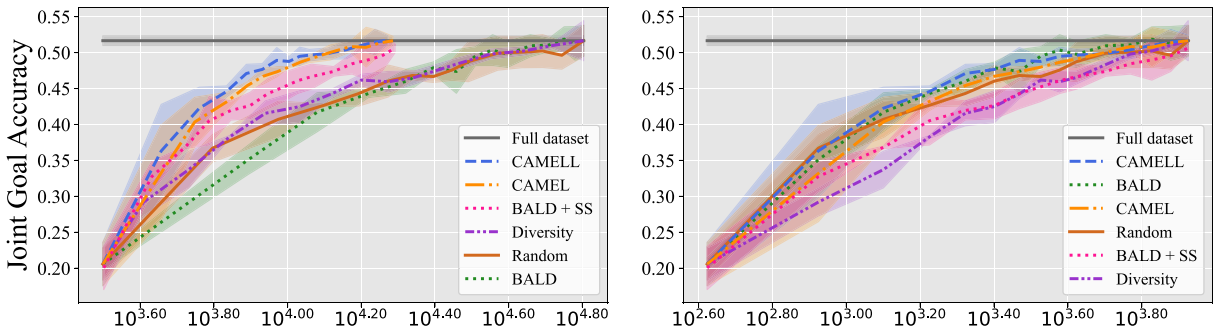
To create a dataset $\mathcal{D}$, annotators usually provide relevant values for the `domain-slot` pairs they believe are present in the user's utterance at every turn t. Subsequently, a handcrafted rule-based tracker considers the previous state $\mathcal{B}_{i,t-1}$, the semantic actions present in the system utterance and the values provided by the annotator to generate complete dialogue states for each turn (Budzianowski et al., 2018). However, this approach has several drawbacks. Firstly, rule-based trackers tend to be imprecise and necessitate redevelopment for each new application, making it less versatile. Secondly, it may not use the time of human annotators efficiently, as the learning model could potentially predict the state for a substantial part of the dialogue accurately. Lastly, there is the risk of human annotators inadvertently overlooking slots in the user input, which could result in incomplete data.

4.5.1 Learning Model

To apply *CAMEL* to the dialogue belief tracking problem, we use the CE-SetSUMBT (Calibrated Ensemble – SetSUMBT) model (van Niekerk et al., 2021), a model which produces well-calibrated uncertainty estimates, important for CAMEL. The CE-SetSUMBT model consists of 10 ensemble members, requiring 1000 GPU hours to fully train. Approximately 45% of this time is utilized for training the ensemble, 45% for training the *noisy* model, and 10% for training the confidence estimators. In addition, we integrate the post-hoc uncertainty learning using a Dirichlet meta-model approach (Shen et al., 2023), described in Section 3.5, into SetSUMBT.

4.5.2 Datasets

In order to test our proposed approach, we utilize the multi-domain task-oriented dialogue dataset MultiWOZ 2.1 (Eric et al., 2020; Budzianowski et al., 2018) and its manually corrected test set provided in MultiWOZ 2.4 (Ye et al., 2022). In our experiments, we regard MultiWOZ 2.1 as a dataset with substantial label noise (Eric et al.,



(a) Number of labels

(b) Number of dialogues

Figure 6: JGA of the CE-SetSUMBT model using different active learning approaches, on the MultiWOZ 2.1 test set, as a function of (a) the number of labels and (b) the number of dialogues, with 95% conf. int.

2020; Zang et al., 2020; Ye et al., 2022), and the test set of MultiWOZ 2.4 a dataset with accurate labels.

4.5.3 Implementation Details

The latent dimension of the intra- and inter-category encoders and feature transformation layer is 16. During training of the label confidence estimation model (Section 3.3.2), to avoid overfitting, we improve the calibration of this model by deploying binary label smoothing loss (Szegedy et al., 2016), temperature scaling and noisy training using Gaussian noise (An, 1996).

For the *seed* dataset (Section 3) we randomly select 5% of dialogues on which we train the initial SetSUMBT model. The other dialogues in the dataset are treated as the unlabeled pool. At each update step another 5% of the data are selected to be labeled. At each point where we require expert labels, we take the original labels provided in the dataset to simulate a human annotator.

4.5.4 Evaluation

As the main metric for our experiments, we use joint goal accuracy (JGA) (Henderson et al., 2014). We further include the joint goal expected calibration error (ECE) (Guo et al., 2017; van Niekerk etal., 2020), which measures the calibration of the model. In terms of measuring efficiency of each method, we examine JGA as a function of the number of expert provided labels. In order to assess the quality of the corrected dataset, we measure the JGA of models trained on a noisy dataset, with and without the proposed label correction.

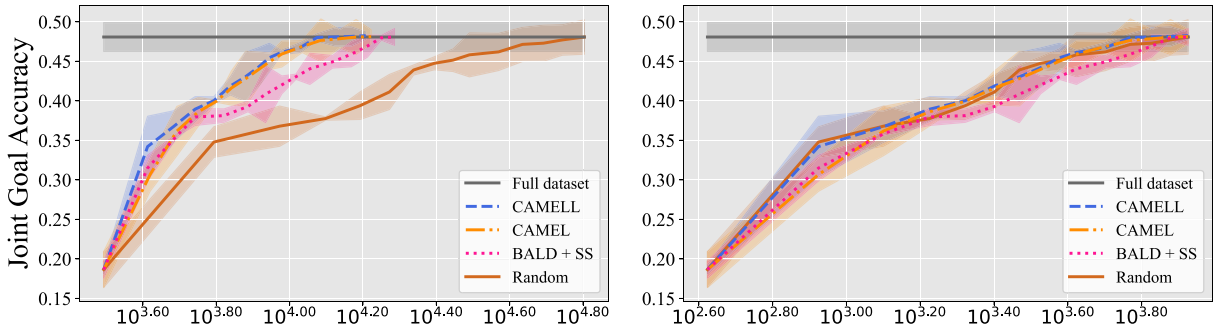
4.5.5 Dialogue Diversity Baseline

We include an additional dialogue diversity baseline, aiming to obtain labels for dialogues geometrically dissimilar from those in the labeled pool, thus ensuring data space coverage. This diversity strategy proposed by Xie et al. (2018) assesses similarity based on vector embeddings of the candidate dialogue versus labeled dialogues. We adapt this approach by employing RoBERTa model embeddings (Liu et al., 2019), fine-tuned in an unsupervised fashion, on the MultiWOZ dialogues.

4.5.6 Results

As shown in Figure 6a, our proposed CAMEL framework requires significantly fewer labels to reach performance levels comparable to those of the baseline methods. This indicates that CAMEL is more efficient in learning dialogue belief tracking than the baseline strategies. It is important to note that all approaches requires the same number of unlabeled dialogues (see Figure 6b). It also highlights the role played by CAMEL's confidence estimates in guiding the active learning process. This conclusion is supported by the lower calibration error of CAMEL's confidence estimates, as reported in Table 1.

Further, we observe in Figure 7a–7b that similar results can be achieved using a computationally efficient uncertainty estimation technique such as the post-hoc Dirichlet meta model, described in Section 3.5, applied to the SetSUMBT model. It should be noted that the comparatively lower joint goal accuracy of this model can be attributed to its singular SetSUMBT model configuration.



(a) Number of labels

(b) Number of dialogues

Figure 7: JGA of the Dirichlet Meta SetSUMBT model using different active learning approaches, on the MultiWOZ 2.1 test set, as a function of (a) the number of labels and (b) the number of dialogues, with 95% conf. int.

An ensemble of models consistently achieves an accuracy that is 2 to 3 percentage points higher.

4.6 Label Correction

To assess the quality of the corrected labels generated by our proposed label correction method (Section 3.4), we trained two distinct tracking models, CE-SetSUMBT and TripPy (Heck et al., 2020), using both the original MultiWOZ 2.1 dataset and various autocorrected datasets (live, online, offline, and semi-offline). The evaluation was conducted on both the noisy MultiWOZ 2.1 test set and the manually corrected MultiWOZ 2.4 test set. The selected tracking models represent the two major non-generative approaches to dialogue state tracking: a pick-list-based approach (SetSUMBT) and a span-prediction approach (TripPy).

4.6.1 Results

In Table 2, we present the JGA of the CE-SetSUMBT models on two test sets: the (noisy) MultiWOZ 2.1 test set and the (manually corrected) MultiWOZ 2.4 test set.[4] Overall, results show the same trend both for CE-SetSUMBT and TripPy: On the MultiWOZ 2.1 test set, the models do not show statistically significant improvements, which is unsurprising given that the MultiWOZ 2.1 test set contains errors and, therefore, cannot adequately assess the impact of label correction. In contrast, on the MultiWOZ 2.4 test set, we observe significant improvements for both offline and online label correction methods for both belief state trackers. This demonstrates that

Model	Label Corr. Setup	MultiWOZ 2.1	MultiWOZ 2.4
CE-SetSUMBT	None	51.79	61.63
	Live	32.48	37.32
	Online	52.85	**63.35**
	Offline	52.83	**63.32**∗
	Semi-offline	52.69	63.12
TripPy	None	55.28	64.45
	Online	56.17	**66.13**
	Offline	56.11	**66.02**∗
	Semi-offline	55.85	65.82

Table 2: Comparison of JGA of trackers trained with and without label corrections. The label corrections can be obtained using a SetSUMBT model trained on the full MultiWOZ 2.1 dataset, trained using CAMEL, or trained using CAMELL. ∗ indicates significant difference on 95% conf. int.

the datasets resulting from online and offline label correction are of significantly higher quality.

The semi-offline method fails to produce significant improvements. We hypothesise that the model trained using CAMEL has already acquired similar error patterns to those commonly made by human annotators. The live label correction setup results in a low-quality dataset, which we attribute to the model's inherent inability to correct data selected through active learning. At this stage, the model lacks the capability to make accurate predictions for these instances.[5]

Although the label validation stage of CAMELL does not yield a statistically significant improvement in the active learning setting, it produces a model that provides more reliable label correction compared to the CAMEL approach without label validation (see online vs. semi-offline correction

[4]The MultiWOZ 2.4 validation set was never used during training.

[5]This method is not examined for TripPy, as we do not expect it to behave differently.

Error Type	Conversation	MultiWOZ 2.1 Labels and Corrections
I	*User:* I would like to find a place that serves **moderately** priced **Chinese** food.	{Restaurant: {Food: Chinese, (95%) Price: Moderate, (94%) Day: Tuesday, (11%) Day: not_mentioned}} (72%)
II	*User:* I feel like going to a **nightclub**. *System:* Okay, the **Soul Tree Nightclub** is a popular place. Would you like the address or phone number? *User:* I will appreciate that.	{Attraction: {Type: Night club, (94%) Name: Soul Tree}, (53%) Hotel: {Name: Sou, (14%) Name: not_mentioned}} (34%)
III	*User:* I need a train leaving on **Friday** and I want to get there by **21 : 30**. Leaving **Broxbourne**.	{Train: {Dept.: Broxbourne, (94%) Day: Friday, (95%) Arrive by: 21:20, (1%) Arrive by: 21:30}} (83%)

Table 3: Examples of three common types of annotation errors in the MultiWOZ 2.1 dataset detected and corrected by CAMELL, (I) hallucinated annotations, (II) multi-annotation and (III) erroneous annotation. For each, we provide the confidence scores of the labels and the corrections proposed by the model. Incorrect labels are marked in red and the proposed corrections in blue.

in Table 2). While CAMELL does not generate labels of higher quality than those produced by the offline label correction approach, it facilitates the creation of a clean dataset with fewer labels, thereby reducing human effort.

An important take-away message is: if all labels in the dataset are available and active learning is not required, offline label correction can be applied to enhance the dataset's quality. However, if labels are being collected through an active learning process, an online label correction should be applied rather than a semi-offline method, as the label validation component enables the creation of a final dataset of higher quality.

4.6.2 Qualitative Analysis

In our investigation of the improved datasets obtained from offline label correction, we identified three prevalent label errors, which our approach successfully rectifies, as exemplified in Table 3. (I) Hallucinated annotations, where the annotator assigns labels not present in the dialogue context, (II) Multi-annotation, the case of assigning multiple labels to the same piece of information, and (III) Erroneous annotation, the situation where an incorrect label is assigned based on the context. These instances underscore the efficacy of our label validation model in minimising the propagation of errors into the dataset.

5 Conclusion

We propose CAMEL, a novel active learning approach that integrates self-supervision, with the goal of minimizing the reliance on labeled data in addressing sequential multi-output labeling problems. Initially, we applied CAMEL to a generative language modeling task in an idealized setting, specifically focusing on machine translation. Subsequently, in a more realistic setting focused on the dialogue belief tracking task, we demonstrated that our approach significantly outperforms baseline methods in terms of robustness and data efficiency.

Additionally, we introduce a methodology for automated dataset correction. Our experiments confirm that our label correction method enhances the overall quality of a dataset. We demonstrate that CAMELL (with label validation) is capable of producing high-quality datasets with a fraction of the human annotation required, through online label correction, thereby highlighting the importance of the label validation component for this task.

Finally, it is important to note that while many presented experiments used ensembles to establish comparisons, we have also provided a mechanism for confidence estimation and active learning that *does not* utilize ensembles and thus is more environmentally friendly.

We believe that this work has far-reaching implications. Firstly, it underscores the indispensable role of uncertainty estimation in learning models. Secondly, the versatility of CAMEL opens up possibilities for its application across diverse sequential multi-output labeling problems, such as entity-relation extraction or weather forecasting. Thirdly, it demonstrates that, in principle, dataset deficiencies can be addressed via data-driven approaches, circumventing the need for extensive manual or rule-based curation. This is particularly pertinent considering the prevailing belief that undesirable outcomes produced by NLP models are inherently linked to the training datasets and cannot be rectified algorithmically (Eisenstein, 2019, 14.6.3).

Looking ahead, we anticipate that refining the process of generating *noisy* datasets could result in a model capable of not only identifying label noise but also filtering out biases, false premises, and misinformation.

Acknowledgments

This work was made possible through the support of the Alexander von Humboldt Foundation, provided within the Sofja Kovalevskaja Award, the European Research Council (ERC) under the Horizon 2020 research and innovation program (grant no. STG2018 804636), and the Ministry of Culture and Science of North Rhine-Westphalia within the Lamarr Fellow Network. Computational resources were provided by the Centre for Information and Media Technology at Heinrich Heine University Düsseldorf and Google Cloud. We thank the anonymous reviewers for their insightful comments and suggestions, particularly for encouraging us to develop a more computationally efficient approach to uncertainty estimation. We also thank Andrey Malinin for early discussions that inspired us to broaden our perspective beyond dialogue state tracking, as well as Prof. Joseph van Genabith for his valuable insights regarding the machine translation setting.

References

Guozhong An. 1996. The effects of adding noise during backpropagation training on a generalization performance. *Neural Computation*, 8(3):643–674. https://doi.org/10.1162/neco.1996.8.3.643

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)*.

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, Ann Arbor, Michigan. Association for Computational Linguistics.

W. H. Beluch, T. Genewein, A. Nurnberger, and J. M. Kohler. 2018. The power of ensembles for active learning in image classification. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA. IEEE Computer Society. https://doi.org/10.1109/CVPR.2018.00976

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 Conference on Machine Translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*. Association for Computational Linguistics. https://doi.org/10.18653/v1/W17-4717

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1547

David A. Cohn, Zoubin Ghahramani, and Michael I. Jordan. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research (JAIR)*, 4:129–145. https://doi.org/10.1613/jair.295

David R. Cox. 1958. The Regression Analysis of Binary Sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 20(2):215–232. https://doi.org/10.1111/j.2517-6161.1958.tb00292.x

Sanjoy Dasgupta and Daniel Hsu. 2008. Hierarchical sampling for active learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 208–215. Association for Computing Machinery. https://doi.org/10.1145/1390156.1390183

Shrey Desai and Greg Durrett. 2020. Calibration of pre-trained transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 295–302. https://doi.org/10.18653/v1/2020.emnlp-main.21

Jacob Eisenstein. 2019. *Introduction to Natural Language Processing*. MIT Press.

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, Marseille, France. European Language Resources Association.

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 International Conference on Machine Learning*, volume 3, pages 1651–1660.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep Bayesian active learning with image data. In *International Conference on Machine Learning*, pages 1183–1192. PMLR.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1321–1330.

Ting Han, Ximing Liu, Ryuichi Takanabu, Yixin Lian, Chongxuan Huang, Dazhen Wan, Wei Peng, and Minlie Huang. 2021. MultiWOZ 2.3: A multi-domain task-oriented dialogue dataset enhanced with annotation corrections and co-reference annotation. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 206–218. Springer. https://doi.org/10.1007/978-3-030-88483-3_16

Michael Heck, Nurul Lubis, Benjamin Ruppik, Renato Vukovic, Shutong Feng, Christian Geishauser, Hsien-chin Lin, Carel van Niekerk, and Milica Gasic. 2023. ChatGPT for zero-shot dialogue state tracking: A solution or an opportunity? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 936–950, Toronto, Canada. Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-short.81

Michael Heck, Nurul Lubis, Carel van Niekerk, Shutong Feng, Christian Geishauser, Hsien-Chin Lin, and Milica Gašić. 2022. Robust dialogue state tracking with weak supervision and sparse data. *Transactions of the Association for Computational Linguistics*, 10:1175–1192. https://doi.org/10.1162/tacl_a_00513

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. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.sigdial-1.4

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014. The second dialog state tracking challenge. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 263–272, Philadelphia, PA, U.S.A. Association for Computational Linguistics. https://doi.org/10.3115/v1/W14-4337

Neil Houlsby, Ferenc Huszar, Zoubin Ghahramani, and Máté Lengyel. 2011. Bayesian active learning for classification and preference learning. *arXiv preprint arXiv:1112.5745 Version 1*.

Junjie Hu and Graham Neubig. 2021. Phrase-level active learning for neural machine translation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 1087–1099, Online. Association for Computational Linguistics.

Andrea Iovine, Pasquale Lops, Fedelucio Narducci, Marco de Gemmis, and Giovanni Semeraro. 2022. An empirical evaluation of active learning strategies for profile elicitation in a conversational recommender system. *Journal of Intelligent Information Systems*, 58(2):337–362. https://doi.org/10.1007/s10844-021-00683-4

Yang Jiao, Shahram Latifi, and Mei Yang. 2019. Self error detection and correction for noisy labels based on error correcting output code in convolutional neural networks. In *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0311–0316. https://doi.org/10.1109/CCWC.2019.8666460

Ulf Johansson, Tuve Lofstrom, and Lars Niklasson. 2007. The importance of diversity in neural network ensembles - An empirical investigation. In *2007 International Joint Conference on Neural Networks*, pages 661–666. https://doi.org/10.1109/IJCNN.2007.4371035

Taejong Joo, Uijung Chung, and Min-Gwan Seo. 2020. Being Bayesian about categorical probability. In *International conference on machine learning*, pages 4950–4961. PMLR.

Ananya Kumar, Tengyu Ma, and Percy Liang. 2020. Understanding self-training for gradual domain adaptation. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5468–5479. PMLR.

Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *International Conference on Machine Learning (ICML) 2013 Workshop: Challenges in Representation Learning (WREPL)*.

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. In *International Conference on Learning Representations (ICLR)*.

Xiaoya Li, Jingrong Feng, Yuxian Meng, Qinghong Han, Fei Wu, and Jiwei Li. 2020b. A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5849–5859. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.519

Zhaojiang Lin, Bing Liu, Seungwhan Moon, Paul Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou Yu, Andrea Madotto, Eunjoon Cho, and Rajen Subba. 2021. Leveraging slot descriptions for zero-shot cross-domain dialogue state tracking. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Ming Liu, Wray Buntine, and Gholamreza Haffari. 2018. Learning to actively learn neural machine translation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 334–344, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/K18-1033

Xin Liu, Shaoxin Li, Meina Kan, Shiguang Shan, and Xilin Chen. 2017. Self-error-correcting convolutional neural network for learning with noisy labels. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 111–117. https://doi.org/10.1109/FG.2017.22

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 Version 1*.

Carel van Niekerk, Michael Heck, Christian Geishauser, Hsien-chin Lin, Nurul Lubis, Marco Moresi, and Milica Gašić. 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. https://doi.org/10.18653/v1/2020.findings-emnlp.277

Carel van Niekerk, Andrey Malinin, Christian Geishauser, Michael Heck, Hsien-chin Lin, Nurul Lubis, Shutong Feng, and Milica Gašić. 2021. Uncertainty measures in neural belief tracking and the effects on dialogue policy performance. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana,

Dominican Republic. Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.emnlp-main.623

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210. https://doi.org/10.1109/ICASSP.2015.7178964

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics. https://doi.org/10.3115/1073083.1073135

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1).

Jesse Read, Luca Martino, Pablo M. Olmos, and David Luengo. 2015. Scalable multi-output label prediction: From classifier chains to classifier trellises. *Pattern Recognition*, 48(6):2096–2109. https://doi.org/10.1016/j.patcog.2015.01.004

Scott E. Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. 2015. Training deep neural networks on noisy labels with bootstrapping. In *The International Conference on Learning Representations (ICLR) (Workshop)*.

Ricardo Rei, Craig Stewart, Ana C. Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702. Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.213

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252. https://doi.org/10.1007/s11263-015-0816-y

Max Ryabinin, Andrey Malinin, and Mark Gales. 2021. Scaling ensemble distribution distillation to many classes with proxy targets. *Advances in Neural Information Processing Systems*, 34:6023–6035.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison, Department of Computer Sciences.

Maohao Shen, Yuheng Bu, Prasanna Sattigeri, Soumya Ghosh, Subhro Das, and Gregory Wornell. 2023. Post-hoc uncertainty learning using a dirichlet meta-model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 9772–9781. https://doi.org/10.1609/aaai.v37i8.26167

Yanyao Shen, Hyokun Yun, Zachary Lipton, Yakov Kronrod, and Animashree Anandkumar. 2017. Deep active learning for named entity recognition. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*. Vancouver, Canada. Association for Computational Linguistics. https://doi.org/10.18653/v1/W17-2630

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A. Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems*, volume 33, pages 596–608.

Matthias Sperber, Graham Neubig, Satoshi Nakamura, and Alex Waibel. 2016. Optimizing computer-assisted transcription quality with iterative user interfaces. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*,

pages 1986–1992, Portorož, Slovenia. European Language Resources Association (ELRA).

Pei-Hao Su, Milica Gašic, and Steve Young. 2018. Reward estimation for dialogue policy optimisation. *Computer Speech and Language*, 51:24–43. https://doi.org/10.1016/j.csl.2018.02.003

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2022. Multi-task pre-training for plug-and-play task-oriented dialogue system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4661–4676, Dublin, Ireland. Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.acl-long.319

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. 2015. Training convolutional networks with noisy labels. In *3rd International Conference on Learning Representations, ICLR 2015 (Workshop)*.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. 2017. Inception-v4, Inception-ResNet and the impact of residual connections on learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pages 4278–4284. AAAI Press. https://doi.org/10.1609/aaai.v31i1.11231

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826. https://doi.org/10.1109/CVPR.2016.308

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401 Version 1*.

Blaise Thomson and Steve Young. 2010. Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Computer Speech and Language*, 24(4):562–588. https://doi.org/10.1016/j.csl.2009.07.003

Neeraj Vashistha, Kriti Singh, and Ramakant Shakya. 2022. Active learning for neural machine translation. *arXiv preprint arXiv:2301.00688 Version 1*. https://doi.org/10.2139/ssrn.4316582

Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. 2015. Learning from massive noisy labeled data for image classification. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2691–2699. https://doi.org/10.1109/CVPR.2015.7298885

Kaige Xie, Cheng Chang, Liliang Ren, Lu Chen, and Kai Yu. 2018. Cost-sensitive active learning for dialogue state tracking. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 209–213, Melbourne, Australia. Association for Computational Linguistics. https://doi.org/10.18653/v1/W18-5022

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. 2020. Self-training with noisy student improves imagenet classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695. https://doi.org/10.1109/CVPR42600.2020.01070

Fanghua Ye, Jarana Manotumruksa, and Emine Yilmaz. 2022. MultiWOZ 2.4: A multi-domain task-oriented dialogue dataset with essential annotation corrections to improve state tracking evaluation. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 351–360, Edinburgh, UK. Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.sigdial-1.34

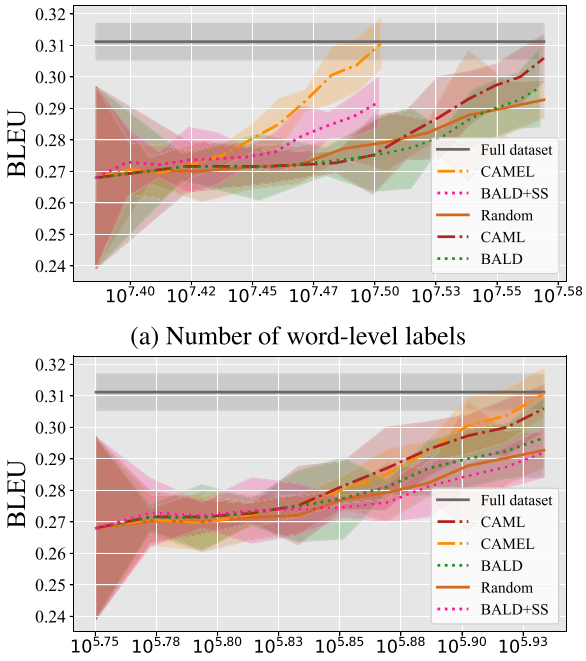
Steve Young, Milica Gašic, Simon Keizer, François Mairesse, Jost Schatzmann, Blaise Thomson, and Kai Yu. 2010. The hidden information state model: A practical framework for POMDP-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174. https://doi.org/10.1016/j.csl.2009.04.001

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–IV–152. https://doi.org/10.1109/ICASSP.2007.367185

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2: A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.nlp4convai-1.13

A BLEU Scores for Translation Experiments



(a) Number of word-level labels

(b) Number of complete translations

Figure 8: BLEU score of the T5 translation model using different active learning approaches on the WMT17 DE-EN test set, as a function of (a) the number of word-level labels and (b) the number of complete translations, with 95% conf. int.