DAMsL: A meta-learning based approach for dialogue state tracking

Stanford CS224N Custom Project; **Grading Option #3**

Avanika Narayan
Department of Computer Science
Stanford University
avanikan@stanford.edu

Josh Hedtke
Department of Computer Science
Stanford University
jhedtke@stanford.edu

Abstract

Recent end-to-end trainable neural network based approaches have demonstrated state-of-the-art results on dialogue state tracking (DST) for task-oriented dialogue agents. In order to perform optimally, these models require access to large, annotated datasets. However, in real-world settings, task-oriented datasets are small and hard to acquire. In this work, we present a framework for learning a generalizable dialogue state tracking model (DAMsL) which can quickly adapt to unseen domains in data constrained settings. We use key principles from model-agnostic methods for meta-learning (MAML) to train a model which learns a set of optimal model parameters that can easily be fine-tuned on a new, low-resource target domain. Given a set of learning tasks (slot-value pair prediction for different domains), DAMsL is trained such that it can effectively generalize to a new unseen domain given only a few training examples. Empirically, we show that DAMsL outperforms other training procedures in various low-resource settings. Moreover, we demonstrate that when compared with other generalization learning frameworks such as multi-task learning, DAMsL is less prone to over-fitting on held-out low-resource task domains.

1 Introduction

Dialogue state tracking (DST) is an integral part of dialogue management in task-oriented dialogue systems. The task itself involves estimating the beliefs of possible user goals at every dialogue turn. It does so by keeping track of a set of slots-attribute pairs (i.e. dialogue state). The objective of the DST is to estimate the correct slot-value pairs based on a number of features including conversation history. Generally, DSTs are designed to fit a specific application domain or task. However, recent efforts have been made towards developing scalable, domain-agnostic DST models [1,6,9]. With the advent of domain-agnostic models, the question arises as to how generalizable these models are and whether they are capable of transferring knowledge to new domains. Given that collecting a large-scale task-oriented dataset for a new domain is expensive and time-consuming, designing models which are capable of generalizing to unseen domains with access to only a few training samples is of primary interest. While there exists a large body of work related to improving the computational efficiency of DST models [7,8], there have been relatively few efforts towards few-shot learning for dialogue state tracking. In [9], Wu et al., explore the efficacy of their TRADE model in zero-shot DST and few-shot DST. The paper indicates that the domain-sharing characteristic of their TRADE model enables it to perform zero-shot DST on unseen domains and adapt in few-shot settings. However, beyond this effort, there exists no other current literature (to the knowledge of the authors) which explores the topic and thus makes it difficult to evaluate the results of the TRADE model in few-shot/zero-shot DST.

While the TRADE model relies on its model architecture to enable few-shot learning, this paper instead casts the problem of knowledge transferability and fast generalization in low-resources

domains as an optimization-based meta-learning problem. Using principles from the model-agnostic meta-learning (MAML) algorithm [2], we propose a generalizable dialogue state tracking model (DAMsL) which is capable of quickly adapting to new domains in data-constrained regimes. The MAML algorithm attempts to build an internal representation that is suitable to many tasks and that has maximal sensitivity to the loss function of new tasks.

Based on the MAML algorithm, we construct an optimization framework which is used to train a flexible, domain-agnostic dialogue state tracking model inspired by HyST [1]. We train DAMsL over a set of meta-tasks which we define as dialogue turns from single-domain dialogues from a set of domains.

The key contributions of our paper are as follows:

- To the knowledge of the authors, DAMsL is the first attempt at using meta-learning to learn a generalizable DST model
- DAMsL has better generalization abilities in low-resource settings than models trained via generalization learning frameworks such as multi-task learning
- We empirically show that on some held-out domains, DAMsL is less prone to overfitting the low-resource task.

2 Related Work

2.1 Domain State Tracking

There exists a number of neural-based approaches to DST. [1] and [6] both propose scalable, domain-agnostic DST models. The DST model that serves as the basis of our work is largely inspired by the Open Vocabulary State tracking (OV ST) model introduced in [1]. We choose to implement this model because of its flexibility and ease of extensibility to new domains: two necessary characteristics for our optimization-based learning framework DAMSL

2.2 Meta-learning

Meta-learning [10,11] is a field of machine learning which tackles the problem of learning how to learn. Recently, [2] introduced model-agnostic meta-learning (MAML) . Unlike previous meta-learning approaches, MAML does not place constraints on allowable model architectures and simply requires that the model can be optimized via gradient descent. Given that the OV ST model from [1] can be optimized using gradient descent, we found MAML to be suitable for our needs. The use of MAML in NLP applications is not expansive. Some examples include using meta-learning for low-resource natural language generation [12], developing personalized dialogue agents [13] and machine translation for low-resource languages [14]. To the best of our knowledge, this is the first attempt in adapting meta-learning to the task of DST.

3 Approach

3.1 Models and Training Frameworks

- Models
  - **ORIGINAL-DST**: DST model with analogous architecture and hyper-parameters as the Open Vocabulary State tracking (OV ST) model in [1].
  - **SUPERVISED-DST**: Modified ORIGINAL-DST model with domain specific hyper-parameters and architecture, trained with full access to all domain-specific dialogue data. We train a Supervised-DST on the Train (TRAIN-DST) and Attraction (ATTRACTION-DST) domains.
- Training Frameworks
  - **MTL-DST**: Train the ORIGINAL-DST using a multi-task learning paradigm with source task data (access to 4/5 domains), then fine-tune on a limited number of samples from the held-out target task.

– **DAMsL**: Meta-learned DST which is trained on 4/5 domain dialogue datasets and fine-tuned on the the same limited number of samples from the held-out target task as for MTL-DST.

ORIGINAL-DST is the base model architecture that all of our training frameworks and settings use. We use the MTL-DST training framework as a baseline for DAMSL and we use the SUPERVISED-DST training settings as an oracle for DAMSL.

3.2 ORIGINAL-DST

We implement a DST (ORIGINAL-DST) from scratch based on [1]. For a user turn u_i in a train example dialogue D_i, ORIGINAL-DST produces $\{1, 2, 3\} -$ gram candidate values c_i^j (jth candidate from ith turn, C_i total candidate values for turn i) from the user utterance and outputs binary predictions for each slot value in the dialogue domain.

We use a biLSTM of hidden size 256 to encode the user utterance, which outputs concatenated vector E_i. We use a unidirectional LSTM with hidden size 512 over past utterances $E_1, ..., E_i$, and output the final encoding Z_i. We use a unidirectional LSTM over dialogue acts $s_i^1, ..., s_i^l$ and output the final encoding A_i with size 64. We then feed $([E_i, Z_i, A_i], c_i^j)$ into a fully connected layer to produce an output of size 256.

We minimize the following loss function:

$$L(D) = \sum_{i=1}^{N} \sum_{k=1}^{T} \sum_{j=1}^{C_i} -y_i^{jk} \log \hat{y_i}^{jk} - (1 - y_i^{jk}) \log(1 - \hat{y_i}^{jk})$$

Where

$$\hat{y_i}^{jk} = P(y_i^{jk} | c_i^j, s_i^k, [E_i, Z_i, A_i])$$

3.3 DAMsL

We implement a meta-learning training framework based on [2] and inspired by the github repositories for [2] and [3]. We implement Algorithm 2 in [2] for few shot supervised learning with a tweak suggested by the Finn et al.

Meta-learning learns to learn. There are two learners: the inner-learner which learns over train examples from within a "task" (T_i). And the meta-learner which learns over T_i as examples. Here, $T_i \in \{\text{attraction}, \text{hotel}, \text{restaurant}, \text{taxi}, \text{train}\} = T$, which are our five dialogue domains.

We use Finn et al.,'s meta-objective, where L is as defined above:

$$\min_{\theta} \sum_{T_i \in T} L_{T_i}(D')_{\theta_i'}$$

And the meta-learning algorithm:

Algorithm 1: Few shot meta-learning

Input: α, β, m, k
Input: T
Result: Meta-optimized parameters θ

Initialize θ;
while *still training* **do**
  sample $T_j \subseteq T$ of size m;
  for $T_i \in T_j$ **do**
    sample D of size k from T_i;
    $\theta_i' = \theta - \alpha \nabla_\theta L_{T_i}(D)$;
    sample D' of size k from T_i;
    cache $L_{T_i}(D')_{\theta_i'}$;
  end
  $\theta = \theta - \beta \nabla_\theta \sum_{T_i \in T} L_{T_i}(D')_{\theta_i'}$;
end

We cache the loss computed with the updated parameters in the inner loop which becomes the term summed for T_i in the meta outer loop. θ_i' is thrown away after each inner loop. In other words, we don't compute the Hessian-vector product (Hvp) in the inner loop in the way that [2]. originally formulated. We do this because [2] tried it and noted this first order approximation performs similarly to computing the Hvp while speeding up training by 33% [2].

4 Experiments

4.1 Data

For our state tracking experiments, we use the MultiWoz2.0 dataset [15]. The MultiWoz2.0 dataset was designed for the purposes of research in task-oriented dialogue modeling and contains dialogue conversations across 7 domains with a total of 37 slots. Benchmarks for the dataset include policy optimization, natural language generation and belief tracking (the task that our paper focuses on). The dataset contains 10K dialogues: 3,406 of which are single-domain dialogues and the remaining 7,032 of which are multi-domain dialogues. Each dialogue contains turn-by-turn metadata which include the ground truth (domain, slot, value) pairs. An example turn annotation from the dataset is as follows:

"turn_label": [["train-destination", "cambridge"], ["train-arriveby", "20:45"]]

For the purposes of this project, we used only single-domain dialogues from the MultiWoz2.0. A detailed breakdown of the number of dialogues per domain can be found in Table 1.

hotel	police	hospital	restaurant	train	attraction	taxi
634	245	287	1308	338	150	434

Table 1: Dataset details

4.2 Evaluation method

The three primary evaluation metrics that we use for assessing the performance of our models are as follows:

- **Avg. slot accuracy**: fraction of slots (aggregated over all turns and dialogues) for which the model predicts the correct slot value.
- **Joint goal accuracy**:fraction of dialog turns for which the values for all slots that are predicted correctly.
- **Slot F1/Precision/Recall**: metrics for calculating the candidate/slot classification accuracy of DAMSL

- True positive: correct candidate/slot classification
- False positive: incorrect candidate/slot classification
- False negative: slot which is not filled by the model, that, according to the ground truth label, should be filled.

4.3 Experimental details

4.3.1 Supervised-DST

For the two domains, Train and Attraction, we trained two domain specific DST models: TRAIN-DST and ATTRACTION-DST. Both models were given full access to all available dialogues for their respective domains. The hyper-parameters and architecture for each of the models was chosen based on the results of a rigorous hyper-parameter search. The architecture + hyper-parameter values can be found in Appendix, Table 4.

- **Hyperparameter optimization experiments**: For our hyperparameter optimization experiment, we performed a random search over 5 hyperparameters: learning rate, model dimension scale (relative to ORIGINAL-DST), batch-size, dropout rate in the fully-connected feed-forward network, and the random seed used to initialize the model. In order to efficiently prune bad initializations, we implemented early-stopping which stopped training the model after total binary-cross-entropy (BCE) loss on the validation set increased for more than 2 epochs. All configurations were trained for a max of 10 epochs.
- **Model training**: Both TRAIN-DST and ATTRACTION-DST were trained using 80% of dialogues in their tasks datasets. At train time, we implemented early-stopping which stopped model training after total binary-cross-entropy loss on the validation set increased for more than 2 epochs. Both models were trained using the Adam Optimization algorithm with a fixed learning rate of 0.001 and a batch size of 128. Details on the dataset used for training can be found in Table 2.

	Train	Validation	Test
Train	270	34	34
Attraction	120	15	15

Table 2: Supervised-DST model training dataset

4.3.2 MTL-DST

- **Model training**: In the case of the attraction task, the MTL-DST was trained using dialogue data from each of the remaining 4 domains (train, hotel, restaurant, taxi). The architecture of the MTL-DST model is the same as the Original-DST with the exception that dropout (with a probability of 0.3) is introduced to the fully-connected feed-forward network. The model was trained using the Adam Optimization algorithm with a fixed learning rate of 0.001 and a batch size of 128. Early-stopping was implemented during training.

  The final hyper-parameter that we controlled for was batch-balancing. In order to ensure that no one domain exerted undue influence in the training of the model, we configured our data loader such that each batch included an equal amount of data points from each of the domains.
- **Finetuning**: In order to test the generalization capabilities of the MTL-DST on our held-out datasets, we fine-tuned the MTL-DST model on a fixed number of dialogues from the held-out domain. During the fine-tuning process, the MTL-DST was trained on the limited training set until evaluation loss increases for more than 3 epochs. Additionally, in order to compensate for a large training set imbalance (very few positive examples) a weighted BCE loss was used in order to up-weight the positive examples.

4.3.3 DAMsL

- **Model training**: We trained DAMsL with an inner batch size of 128 and an outer batch size (k domains chosen for each inner loop) of three. We used Adam for both the model

optimizer and the meta-optimizer. We trained for 30 meta-epochs with early stopping. We employed gradient clipping at the meta optimization step with max norm of five.

- **Finetuning**: We fine-tuned DAMsL exactly the same as MTL-DST to enforce a controlled experiment.

4.4 Results

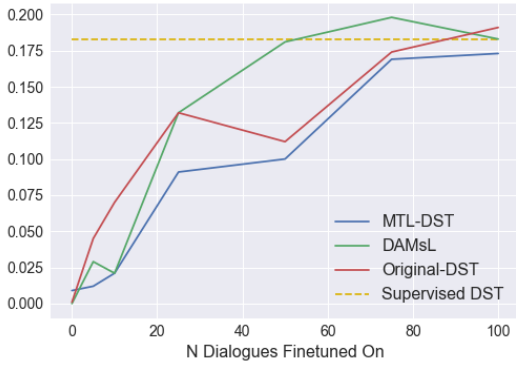

Figure 1: Attraction: F1 score

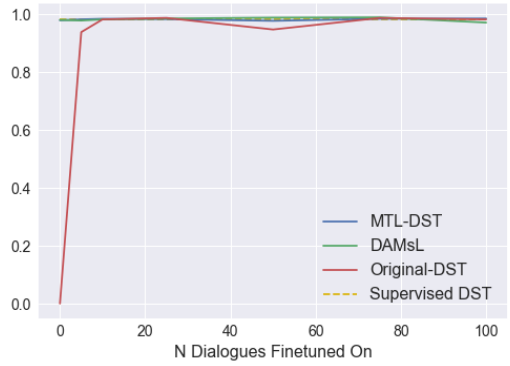

Figure 2: Attraction: Avg. Slot Accuracy

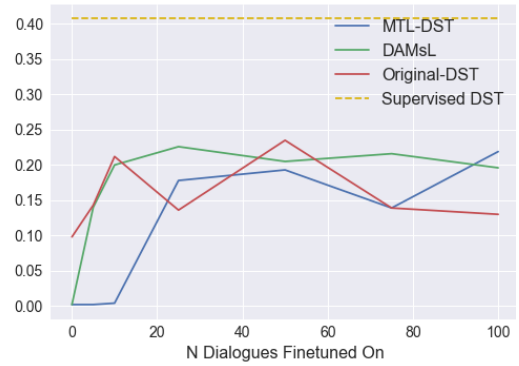

Figure 3: Train: F1 score

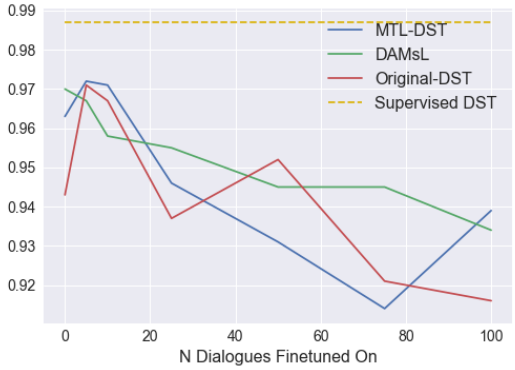

Figure 4: Train: Avg. Slot Accuracy

5 Analysis

5.1 Qualitative Error Analysis

- **Overfilling**: As seen in ex. 3 and 6, the fine-tuned models are prone to slot-overfilling. We hypothesize that this is the result of the positive up-weighting we used during fine-tuning when computing BCE loss.
- **Choice of 2/3-gram over 1-gram**: As seen in ex. 1 and 5, the models may choose candidates which are bi-grams or tri-grams with similar content to the ground truth unigram that fills the slot.
- **Overuse of 'dontcare'**: It was observed that the ORIGINAL-DST was prone to choosing candidates such as 'dontcare' or other non-sensible candidates. We hypothesize that this behavior is due to the fact that the ORIGINAL-DST doesn't have the same pre-training exposure of DAMSL and MTL-DST and is thus more likely to choose unlikely candidates to fill slots.

Ex.	Domain Name	Model Name	# of finetuning examples	Ground truth	Predicted
1	Attraction	MTL-DST	75	21: 'swimmingpool'	21: 'find a swimmingpool'
2	Attraction	ORIGINAL-DST	50	21: 'swimmingpool', 15: 'north'	21: 'dontcare', 15: 'dontcare'
3	Train	MTL-DST	75	14: '13:00	9: 'cambridge', 10: 'sunday', 11: 'cambridge', 14: '21:00'
4	Attraction	DAMSL	50	21: 'swimmingpool', 15: 'north'	15: 'north'
5	Attraction	DAMSL	50	19: 'the fitzwilliam museum'	19: 'the fitzwilliam museum', 21: 'museum'
6	Train	DAMSL	75	10:'thursday', 11:'Bishops stortford	9: 'leaving on thursday', 10:'thursday', 11:'departing from bishops'

Table 3: Slot-filling model outputs

5.2 Reasons for low accuracy

As seen in Figure 4, the slot accuracy of all the trained models decreases as the number of fine-tuning examples increases. We hypothesis that this loss in accuracy is a result of the positive up-weighting we used when computing BCE loss during fine-tuning. The positive up-weighting encouraged the fine-tuned model to fill all domain specific slots at every turn. Thus, the model tended to overfill reducing its average slot accuracy. We tested our hypothesis empirically and found that when trained without positive up-weighting our evaluation metrics increased by approximately 2-fold. Moreover, in the absence of positive up-weighting, the average number of slots filled per turn decreased by 6-fold.

5.3 Performance of DAMsL

Our experiments indicate that DAMSL has better generalization capabilities than MTL-DST. When fine-tuned on both the Train and Attraction domains, the F1 scores of DAMSL consistently outperformed that of the MTL-DST model. Moreover, as seen in Figure 4, the slot-filling performance of the DAMSL was less effected by the loss weighting than MTL-DST or the Original-DST. This suggests that DAMSL 1) may learn a more general representation of what makes a good slot value than MTL-DST and 2) is less prone to over-fitting when fine-tuned on a low-resource target domain.

6 Conclusion

In order to perform optimally, recently introduced end-to-end DST models require access to large, annotated datasets. However, in real-world settings, task-oriented datasets are small and hard to acquire. In this work, we present a framework for learning a generalizable dialogue state tracking model (DAMSL) which can quickly adapt to unseen domains in data constrained settings. Our main contributions include: 1) introducing DAMsL - the first attempt at using meta-learning to learn a generalizable DST model and 2) empirically proving that DAMsL outperforms other training procedures in various low-resource settings and is less prone to overfitting on held-out low-resource task domains.

6.1 Future Work

There are many optimizations for DAMSL we would like to explore. For example, Antoniou et al., propose a few optimizations they claim improve train time and generalization [4]. Some of the

optimizations are obvious, such as using an annealing learning rate instead of fixed for the meta-learner. Others are less so: derivative-order annealing wherein they use the first order approximation for the first N epochs then switch to the Hvp for the remaining epochs and learning the inner learning rate for each parameter on each update step of the inner loop.

We used a fix value of k for the inner loop sample size. We would like to do a hyperparameter search that includes k. We also limit the inner loop to a single gradient step per task. This combined with the fact that we only train on half the examples as our other models/frameworks (because the other have become the train examples for the meta-learner) may limit DAMSL's ability to overfit.

Additionally, we would also like to explore the efficacy of pre-trained embeddings (GloVe, fastText) in learning a more generalizable model initialization.

Acknowledgement

We thank our project mentor, Matt Lamm, for all his support and guidance throughout this project!

References

[1] Rahul Goel, Shachi Paul, and Dilek Hakkani-Tür. Hyst: A hybrid approach for flexible and accurate dialogue state tracking, 2019.

[2] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017.

[3] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml, 2018.

[4] Kun Qian and Zhou Yu. Domain adaptive dialog generation via meta learning. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[5] Hosseini-Asl Xiong Socher Fung Wu, Madotto. Transferable multi-domain state generator for task-oriented dialogue systems. *arXiv*, 2019.

[6] Rahul Goel, Shachi Paul, Tagyoung Chung, Jeremie Lecomte, Arindam Mandal, and Dilek Hakkani-Tur. Flexible and scalable state tracking framework for goal-oriented dialogue systems, 2018.

[7] Tuan Manh Lai, Quan Hung Tran, Trung Bui, and Daisuke Kihara. A simple but effective bert model for dialog state tracking on resource-limited systems, 2019.

[8] Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. Efficient dialogue state tracking by selectively overwriting memory, 2019.

[9] Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. Transferable multi-domain state generator for task-oriented dialogue systems, 2019.

[10] Sebastian Thrun and Lorien Pratt, editors. *Learning to Learn*. Kluwer Academic Publishers, USA, 1998.

[11] Samy Bengio, Yoshua Bengio, Jocelyn Cloutier, and Jan Gecsei. On the optimization of a synaptic learning rule, 1997.

[12] Fei Mi, Minlie Huang, Jiyong Zhang, and Boi Faltings. Meta-learning for low-resource natural language generation in task-oriented dialogue systems, 2019.

[13] Zhaojiang Lin, Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. Personalizing dialogue agents via meta-learning, 2019.

[14] Jiatao Gu, Yong Wang, Yun Chen, Kyunghyun Cho, and Victor O. K. Li. Meta-learning for low-resource neural machine translation, 2018.

[15] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. Multiwoz - a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling, 2018.

A Appendix

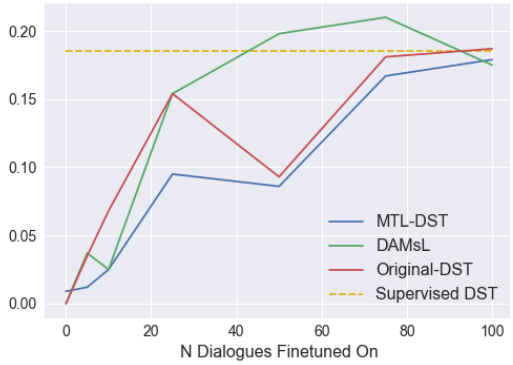

Figure 5: Attraction, Slot Precision

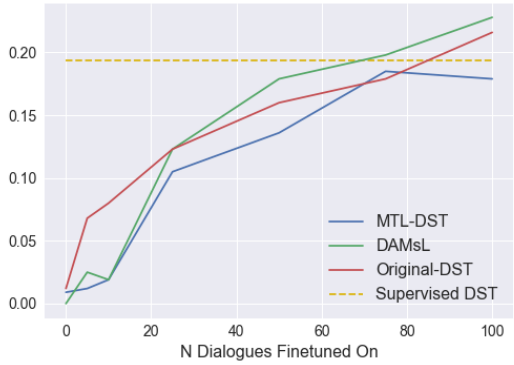

Figure 6: Attraction, Slot Recall

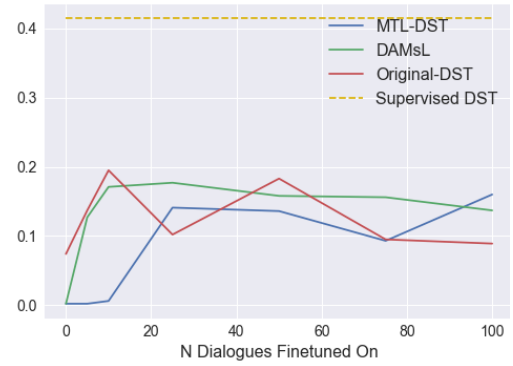

Figure 7: Train, Slot Precision

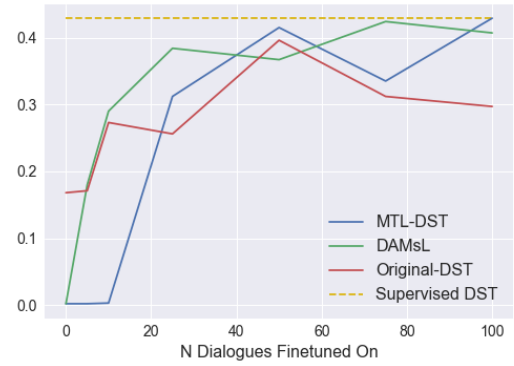

Figure 8: Train, Slot Recall

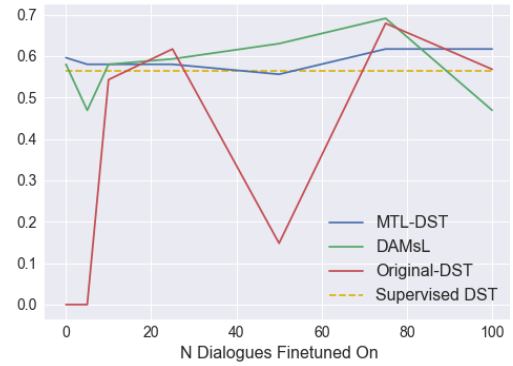

Figure 9: Attraction, Joint Goal Accuracy

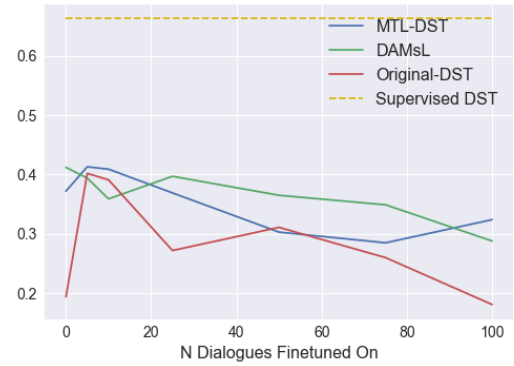

Figure 10: Train, Joint Goal Accuracy

	Learning Rate	Scale	Batch Size	Dropout Prob.	Random Seed
ATTRACTION-DST	.001	0.125	8	0.4	369
TRAIN-DST	.0001	.03125	32	0.1	447

Table 4: Supervised-DST model architectures