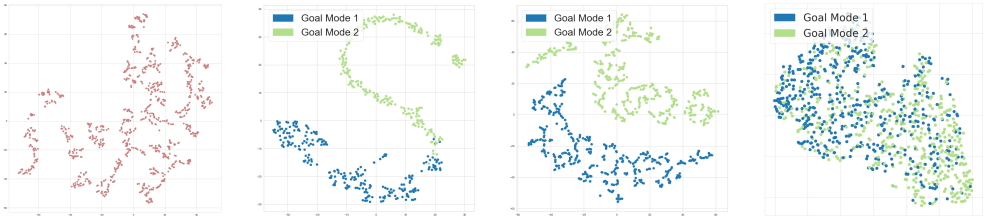


(a) Few-shot classification (b) 2D Navigation interpolated goals (c) 2D Navigation random goals (d) Target Speed random goals

Figure 5: tSNE plots of task embeddings produced in different problem settings. (a) Embeddings of randomly sampled tasks in few-shot image classification. (b-c) Embeddings of tasks with goal locations interpolated between the modes and sampled randomly in 2D Navigation. (d) Randomly sampled goals in target speed environment.

A Modulation Methods

To allow efficient adaptation, the modulation network activates or deactivates units of each network block of the gradient-based meta-learner according to the given target task embedding. We investigated a representative set of modulation operations, including attention-based modulation [16, 30] and feature-wise linear modulation (FiLM) [18].

Attention based modulation [16, 30] has been widely used in modern deep learning models and has proved its effectiveness across various tasks [35, 16, 36, 34]. Inspired by the previous works, we employed attention to modulate the prior model. In concrete terms, attention over the outputs of all neurons (Softmax) or a binary gating value (Sigmoid) on each neuron's output is computed by the model-based meta-learner. These parameters τ are then used to scale the pre-activation of each neural network layer $\mathbf{F}_\theta$, such that $\mathbf{F}_\phi = \mathbf{F}_\theta \otimes \tau$. Note that here $\otimes$ represents a channel-wise multiplication.

Feature-wise linear modulation (FiLM) [18] proposed to modulate neural networks to condition the networks on data from different modalities. We adopt FiLM as an option for modulating our gradient-based meta-learner. Specifically, the parameters τ are divided in to two components τ_γ and τ_β such that for a certain layer of the neural network with its pre-activation $\mathbf{F}_\theta$, we would have $\mathbf{F}_\phi = \mathbf{F}_\theta \otimes \tau_\gamma + \tau_\beta$. It can be viewed as a more generic form of attention mechanism. Please refer to [18] for the complete details.

As shown in the quantitative results (Table 1), using FiLM as a modulation method achieves better results comparing to attention mechanism with Sigmoid or Softmax. We therefore use FiLM for further experiments.

B Few-shot Image Classification

The task of few-shot image classification considers a problem of classifying images into N classes with a small number (K) of labeled samples available. To evaluate our model on this task, we conduct experiments on OMNIGLOT, a widely used handwritten character dataset of binary images. The results are shown in Table 2, demonstrating that our method achieves comparable or better results against state-of-the-art algorithms.

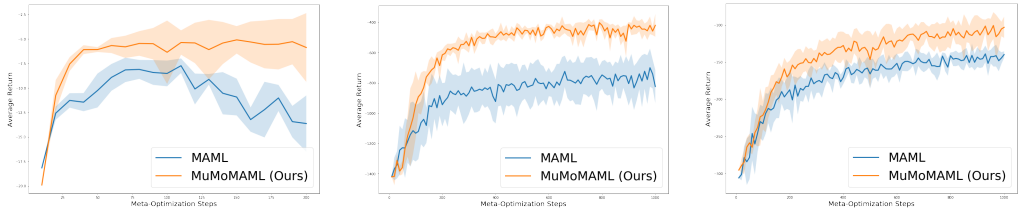
To gain insights to the task embeddings υ produced by our model, we sampled 2000 tasks randomly and employ tSNE to visualize the υ in Figure 5 (a). While we are not able to clearly distinguish the modes of task distributions, we observe that the distribution of the produced embeddings is not uniformly distributed or unimodal, potentially indicating the multimodal nature of this task.

C Implementation Details

For the model-based meta-learner, we used SEQ2SEQ [28] encoder structure to encode the sequence of $\{x, y\}_{k=1,\ldots,K}$ with a bidirectional GRU [3] and use the last hidden state of the recurrent model as the representation for the task. We then apply a one-hidden-layer multi-layer perception (MLP) for

Table 2: 5-way and 20-way, 1-shot and 5-shot classification accuracy on OMNIGLOT Dataset. For each task, the best-performing method is highlighted. MUMOMAML achieves comparable or better results against state-of-the-art few-shot learning algorithms for image classification.

Method	OMNIGLOT			
	5 Way Accuracy (in %)		**20 Way Accuracy (in %)**	
	1-shot	5-shot	1-shot	5-shot
Siamese nets [11]	97.3	98.4	88.2	97.0
Matching nets [31]	98.1	98.9	93.8	98.5
Meta-SGD [13]	99.5	**99.9**	95.9	99.0
Prototypical nets [26]	97.4	99.3	96.0	98.9
SNAIL [15]	99.1	99.8	**97.6**	**99.4**
T-net [12]	99.4	-	96.1	-
MT-net [12]	99.5	-	96.2	-
MAML [5]	98.7	**99.9**	95.8	98.9
MUMOMAML (ours)	**99.7**	**99.9**	97.2	**99.4**



(a) 2D Navigation (b) Target Location (c) Target Speed

Figure 6: Training curves of MUMOMAML and MAML in reinforcement learning environments. The value indicates the performance evaluated after modulation (our model) and 5 gradient steps. The plots demonstrate that our model consistently outperforms MAML from the beginning of training to the end.

each layer in the gradient-based learner's model to generate the set of task-specific parameters τ_i, as described in the previous section. We implemented our models for three representative learning scenarios – regression, few-shot learning and reinforcement learning. The concrete architecture for each task might be different due to each task's data format and nature. We discuss them in the section 3.

D Additional Experimental Details

D.1 Regression

Setups. To form multimodal task distributions, we consider a family of functions including sinusoidal functions (in forms of $A \cdot \sin w \cdot x + b + \epsilon$, with $A \in [0.1, 5.0]$, $w \in [0.5, 2.0]$ and $b \in [0, 2\pi]$), linear functions (in forms of $A \cdot x + b$, with $A \in [-3, 3]$ and $b \in [-3, 3]$) and quadratic functions (in forms of $A \cdot (x - c)^2 + b$, with $A \in [-0.15, -0.02] \cup [0.02, 0.15]$, $c \in [-3.0, 3.0]$ and $b \in [-3.0, 3.0]$). Gaussian observation noise with $\mu = 0$ and $\epsilon = 0.3$ is added to each data point sampled from the target task. In all the experiments, K is set to 5 and L is set to 10. We report the mean squared error (MSE) as the evaluation criterion. Due to the multimodality and uncertainty, this setting is more challenging comparing to [5].

Models and Optimization. In the regression task, we trained a 4-layer fully connected neural network with the hidden dimensions of 100 and ReLU non-linearity for each layer, as the base model for both MAML and MUMOMAML. In MUMOMAML, an additional model with a Bidirectional GRU of hidden size 40 is trained to generate τ and to modulate each layer of the base model. We used the same hyper-parameter settings as the regression experiments presented in [5] and used Adam [10] as the meta-optimizer. For all our models, we train on 5 meta-train examples and evaluate on 10 meta-val examples to compute the loss.

D.2 Reinforcement Learning

Along with few-shot classification and regression, reinforcement learning has been a central problem where meta-learning has been studied [23, 22, 32, 5, 15].

To identify the mode of a task distribution in the context of reinforcement learning, we run the meta-learned prior model without modulation or adaptation to interact with the environment and collect a single trajectory and obtained rewards. Then the collected trajectory and rewards are fed to our model-based meta-learner to compute the task embedding υ and τ. With this minimal amount of interaction with the environment, our model is able to recognize the tasks and modulate the policy network to effectively learn in multimodal task distributions.

The batch of trajectories used for computing the first gradient-based adaptation step is sampled using the modulated model and the batches after that using the modulated and adapted model from the previous update step. We follow the MAML gradient-based adaptation procedure. For the gradient adaptation steps, we use the vanilla policy gradient algorithm [33]. As the meta-optimizer we use the trust region policy optimization algorithm [24].

Models and Optimization. We use embedding model hidden size of 128 and modulation network hidden size of 32 for all environments. The training curves for all environments are presented in Figure 6, which show that our proposed model consistently outperforms MAML from the beginning of training to the end. During optimization we save model parameters and evaluate the model with five gradient update steps every 10 meta update steps. We compute the adaptation curves presented in Figure 3 using the model which achieved the best score after the five gradient updates during training.

2D-Navigation In the 2D navigation environment the goals are sampled with equal probability from one of two bivariate Gaussians with means of $(0.5, 0.5)$ and $(-0.5, -0.5)$ and standard deviation of 0.1. In the beginning of each episode, the agent starts at the origin. The agent's observation is its 2D-location and the reward is the negative distance to the goal. The agent does not observe the goal directly, instead it must learn to navigate there based on the reward function alone. The agent outputs vectors which elements are clipped to the range $[-0.1, 0.1]$ and the agent is moved in the environment by the clipped vector. The episode terminates after 100 steps or when the agent comes to the distance of 0.01 from the goal.

For both MUMOMAML and MAML we sample 20 trajectories for computing the gradient-based adaptation steps and 20 tasks for meta update steps. We use inner loop update step size of 0.1 for MAML and 0.05 for MUMOMAML. We train both methods for 200 meta-optimization steps.

We investigate the behavior of the task embedding network by sampling tasks from the environment and computing a tSNE plot of the task embeddings. A tSNE plot for goals interpolated between the goal modes is presented in Figure 5 (b) and a plot for randomly sampled goals is presented in (c). The tSNE plots show that the structure of the embedding space reflects the goal distribution.

Target Location Target location is a locomotion environment based on the half-cheetah model in the mujoco [29] simulation framework. The environment design follows [5], except for the reward definition. The reward on each time step is

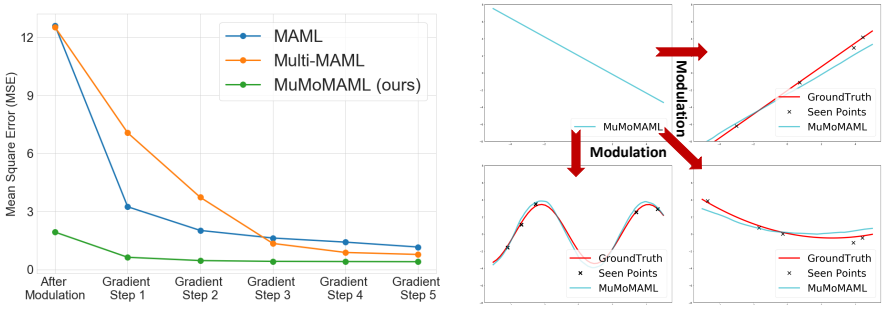
$$R(s) = -1 * abs(x_{torso} - x_{goal}) + \lambda_{control} * \|a\|^2$$

where x_{torso} and x_{goal} are the x-positions of the midpoint of the half-cheetah's torso and the target location respectively, $\lambda_{control} = -0.05$ is the coefficient for the control penalty and a is the action chosen by the agent. The target location is sampled from a distribution consisting of two Gaussians with means of -7 and 7 and standard deviation of 2. The observation is the location and movement state of the joints of the half-cheetah. The episode terminates after 200 steps.

For both MUMOMAML and MAML we sample 20 trajectories for computing the gradient-based adaptation steps and 40 tasks for meta update steps. We use inner loop update step size of 0.05 for both methods. Both methods are trained for 1000 meta-optimization steps.

Target Speed Target speed is another half-cheetah based locomotion environment. The environment design is similar to the target location environment, except the reward is based on achieving target speed. The reward on each time step is

$$R(s) = -1 * abs(v_{agent} - v_{target}) + \lambda_{control} * \|a\|^2$$



(a) Performance upon updates (b) Effect of modulation

Figure 7: **(a)** Comparing the models' performance with respect to the number of gradient updates applied. For MUMOMAML, we report the performance after modulation for gradient step 0. **(b)** A demonstration of the modulation on prior model by our model-based meta-learner. With the FiLM modulation, MUMOMAML can adapt to different priors before gradient-based adaptation.

where v_{agent} and v_{target} are the speed of the head of the half-cheetah and the target speed respectively, $\lambda_{control} = -0.05$ is the coefficient for the control penalty and a is the action chosen by the agent. The target speed is sampled from a distribution consisting of two Gaussians with means of -1 and 1.5 and standard deviation of 0.5. The observation and other environment details are the same as in the target location environment. For training MUMOMAML and MAML same hyperparameters are used in as for target location environment.

A tSNE plot of randomly sampled task embeddings from the target speed environment is presented in Figure 5 (d). The embeddings for tasks from different modes are distributed towards the opposite ends of the tSNE plot, but the modes are not as clearly distinguishable as in the other environments. Also, the modulated policy in the target speed environment achieves lower returns than in other environments as is evident from Figure 3. Notice that in the reinforcement learning setting, the model is not optimized to achieve high returns immediately after the modulation step but only after modulation and one gradient update. After one gradient step MUMOMAML consistently outperforms MAML in target speed as well.

D.3 Few-shot Image Classification

Setups. In the few-shot learning experiments, we used OMNIGLOT, a dataset consists of 50 languages, with a total of 1632 different classes with 20 instances per class. Following [20], we downsampled the images to 28×28 and perform data augmentation by rotating each member of an existing class by a multiple of 90 degrees to form new data points.

Models and Optimization. Following prior works [31, 5], we used the same 4-layer convolutional neural network and applied the same training and testing splits from [5] and compare our model against baselines for 5-way and 20-way, 1-shot and 5-shot classification.

E Additional Experimental Results

E.1 Additional Results for Regression

Figure 7 demonstrates that MUMOMAML outperforms the MAML and Multi-MAML baselines no matter how many gradient steps are performed. Also, MUMOMAML is able to achieve good performance solely based on modulation without any gradient update, showing that our model-based meta-learner is capable of identifying the mode of a multimodal task distribution and effectively modulate the meta-learned prior.

Additional qualitative results for MUMOMAML after modulation are shown in Figure 8 and additional qualitative results for MUMOMAML after adaptation are shown in Figure 9.

E.2 Additional Qualitative Results for Reinforcement Learning

Additional trajectories sampled from the 2D navigation environment are presented in Figure 10.

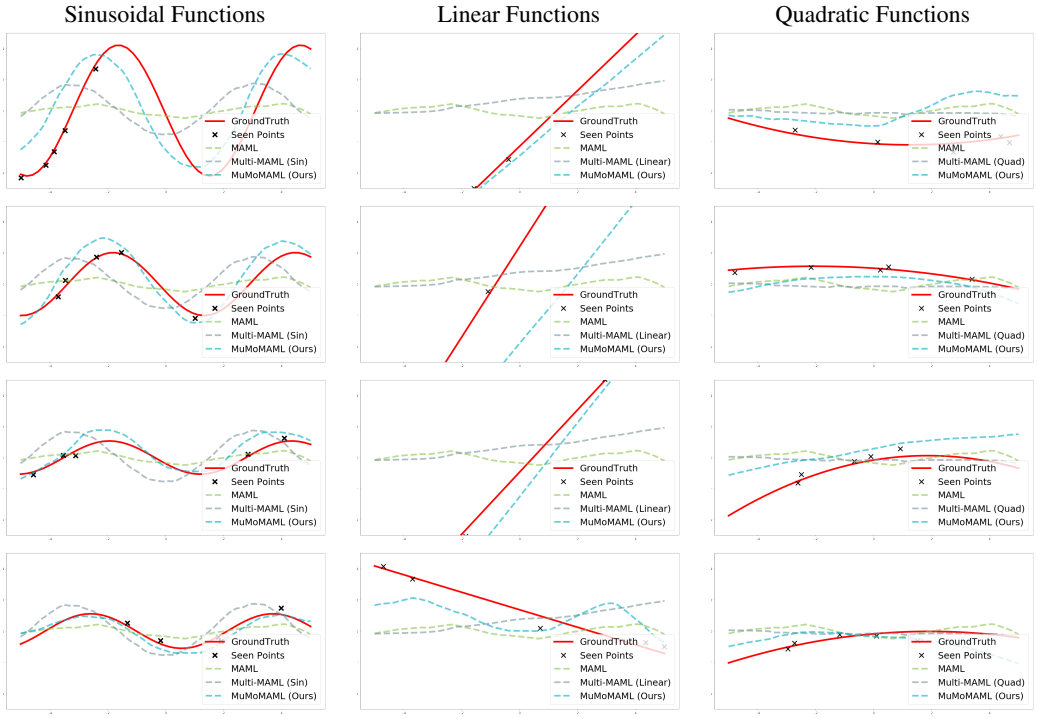


Figure 8: Additional qualitative results of the regression tasks. MUMOMAML **after modulation** vs. other prior models.

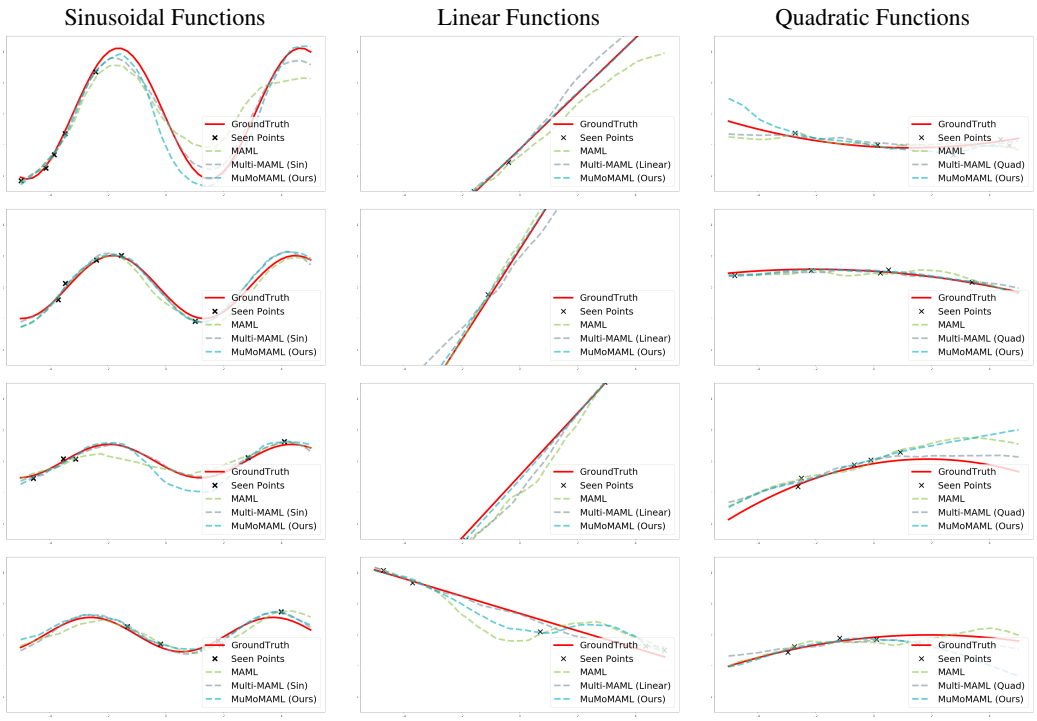


Figure 9: Additional qualitative results of the regression tasks. MUMOMAML **after adaptation** vs. other posterior models.

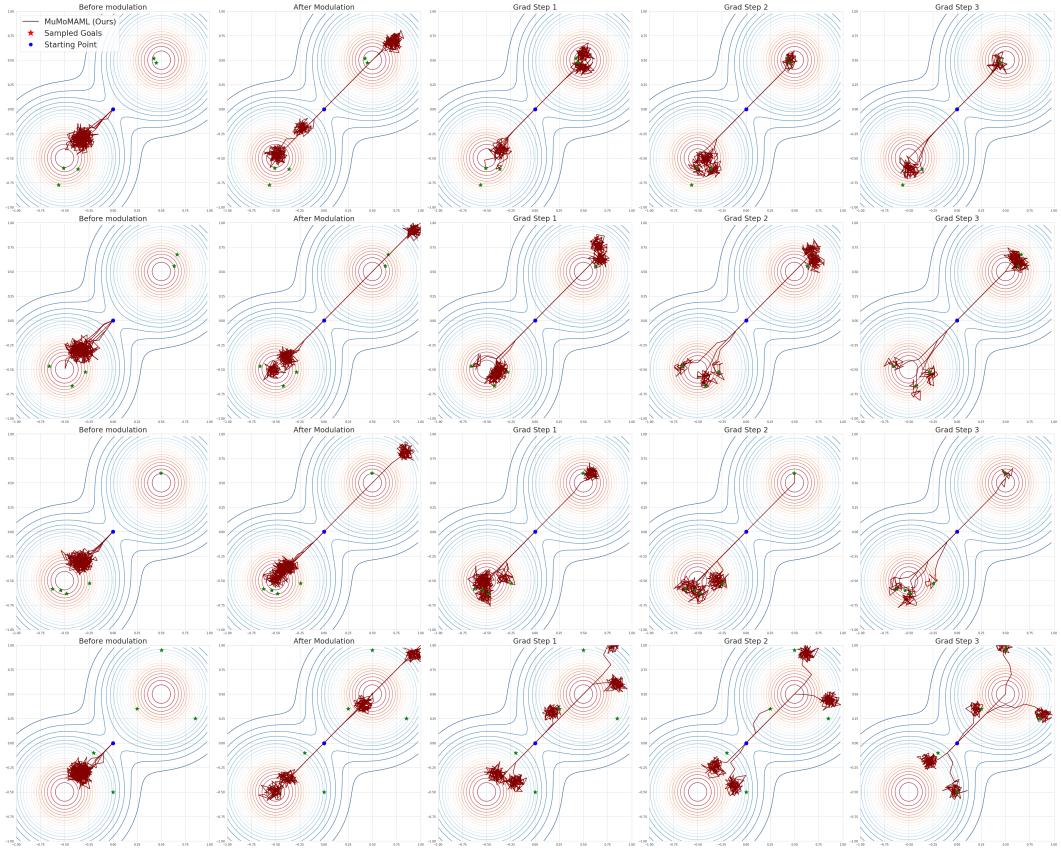


Figure 10: Additional trajectories sampled from the 2D navigation environment with MUMOMAML. The first four rows are with goals sampled from the environment distribution, where MUMOMAML demonstrates rapid adaptation and is often able to locate the goal exactly. On the fifth row, trajectories are sampled with less probable goals. The agent is left farther away from the goals after the modulation step, but the gradient based adaptation steps then steadily recover the performance.