TaskMet: Task-Driven Metric Learning for Model Learning

Dishank Bansal, Ricky T. Q. Chen, Mustafa Mukadam, Brandon Amos

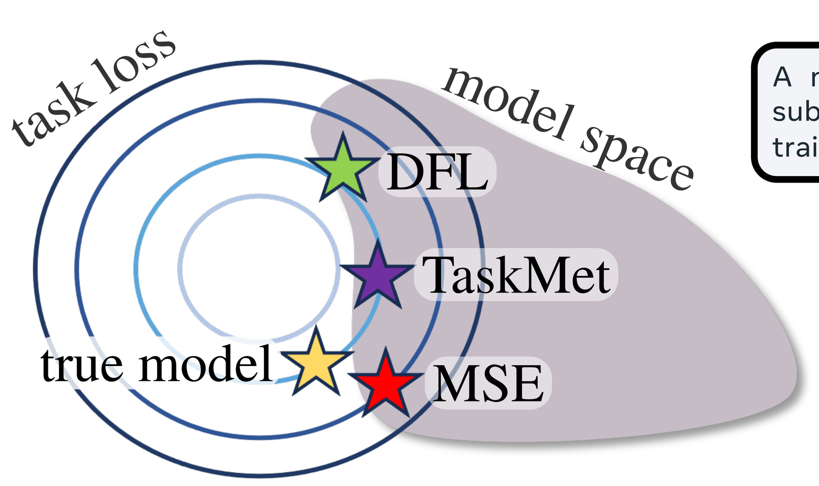
Motivation: training models for downstream tasks

Challenge: models trained with prediction losses may struggle on downstream tasks

Why? objective mismatch, approximation errors, limited capacity, data

Our contribution: a task-driven end-to-end metric learning framework for training prediction models. This provides:

- A method to train models for better performance on the downstream task
- A method to learn a loss function using task information, which is then used to train prediction model



A model trained with MSE may still perform suboptimally on the downstream task. TaskMet trains the model to achieve minimal task loss.

Examples of two-stage settings

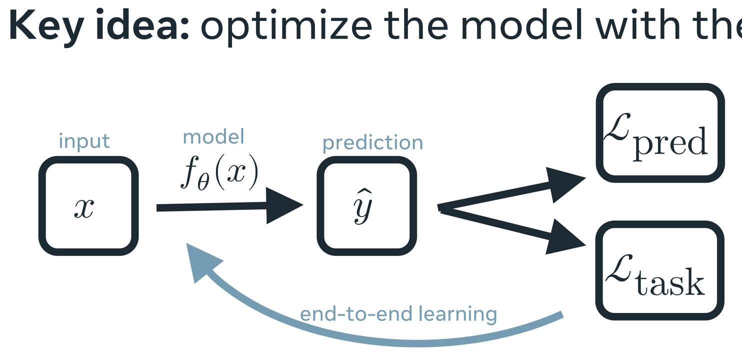
models/predictions	downstream tasks
click-through rates	recommendations
item demands/utilities	budget allocation
MDP state transitions	optimal control
future stock prices	portfolio optimization
next word (LLMs)	chatbots (RLHF)

(MSE, likelihood) $\mathcal{L}_{\text{pred}}(\theta) \neq \mathcal{L}_{\text{task}}(\theta)$

Background: task-based learning

Task-based end-to-end model learning in stochastic optimization. Donti, Amos, and Kolter, NeurIPS 2017.
Decision-Focused Learning for Combinatorial Optimization. Wilder et al., AAAI 2019.
Smart "Predict, then optimize." Elmachtoub and Grigas, Management Science 2022.

Key idea: optimize the model with the task loss

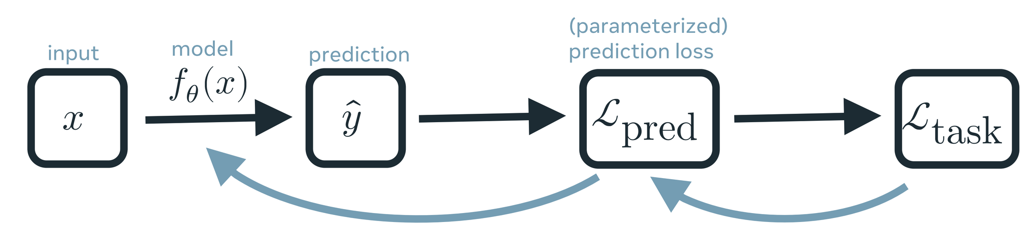


$$\theta^\star := \operatorname*{argmin}_\theta \mathcal{L}_{\text{task}}(\theta) + \alpha \mathcal{L}_{\text{pred}}(\theta)$$

Drawbacks of standard task-based losses

1. The model may **overfit to the task** and be unable to generalize to other tasks
e.g., one task may care about colors while another may care about edges
2. The model may **forget how to predict in the original space**
e.g., the task loss may just care about magnitudes rather than absolute values

Our paper: task-based metric learning (TaskMet)



Insight: use the task signal to shape the metric of a prediction loss, not the model

Why?

1. **Retains the prediction task:** the metric emphasizes the important parts
2. **Generalization:** the metric can be controlled to not deviate too much

Generalized Mahalanobis loss as the prediction loss

We use a generalized Mahalanobis loss parameterized by the metric Λ_ϕ

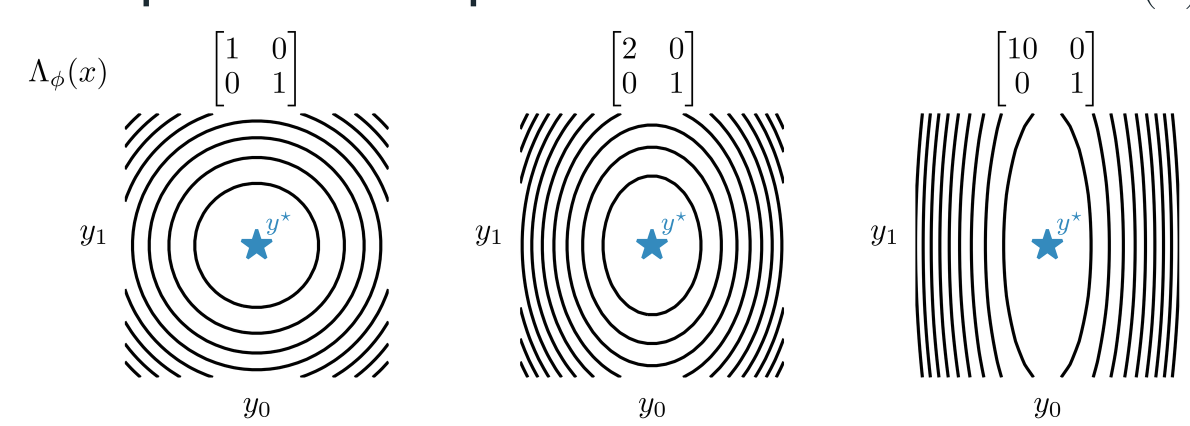
$$\mathcal{L}_{\text{pred}}(\theta,\phi) := \mathbb{E}_{x,y\sim D}\left[\|f_\theta(x)-y\|^2_{\Lambda_\phi(x)}\right] = \mathbb{E}_{x,y\sim D}[(f_\theta(x)-y)^T\Lambda_\phi(x)(f_\theta(x)-y)]$$

The **metric** $\Lambda_\phi(x)$, is a

- positive semi-definitive matrix
- of dim $n{\times}n$, where n is dimension of prediction space
- parameterized by ϕ and conditioned on the input features x

The metric captures:

- **Relative importance of features:** up/down-weighting based on importance
- **Relative importance of samples:** via heteroscedastic metric $\Lambda(x)$



End-to-end metric learning for model learning

- The key idea of the method is to **learn an optimal metric** end-to-end with a given task, which is then used to learn model via prediction loss.
- We use **bilevel optimization** for the metric and model learning:

$$\boxed{\begin{aligned}\phi^\star &:= \operatorname*{argmin}_\phi \mathcal{L}_{\text{task}}(\theta^\star(\phi))\\ \text{where}\quad \theta^\star(\phi) &= \operatorname*{argmin}_\theta \mathcal{L}_{\text{pred}}(\theta,\phi) \implies \nabla_\theta \mathcal{L}_{\text{pred}}(\theta,\phi)\big|_{\theta=\theta^\star(\phi)} = 0\end{aligned}}$$

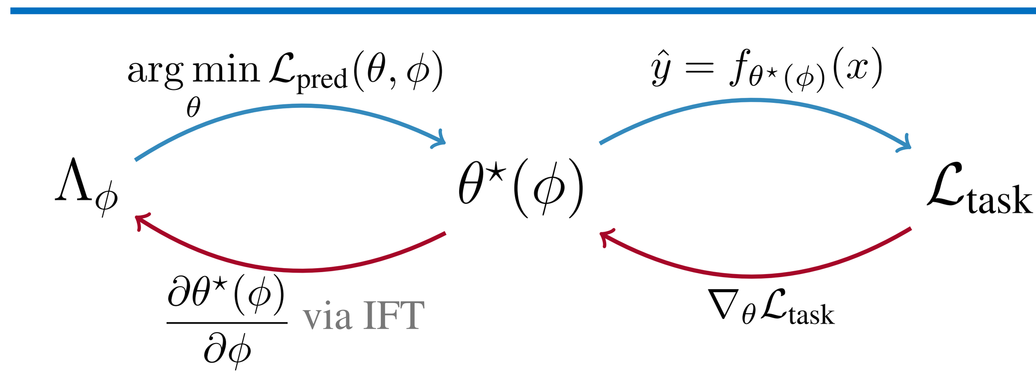
Implicit differentiation for end-to-end metric learning

We need calculate $\nabla_\phi \mathcal{L}_{\text{task}}(\theta^\star(\phi))$, to find $\phi^\star$

$$\nabla_\phi \mathcal{L}_{\text{task}}(\theta^\star(\phi)) = \nabla_\theta \mathcal{L}_{\text{task}}(\theta)\big|_{\theta=\theta^\star(\phi)} \cdot \underbrace{\frac{\partial \theta^\star(\phi)}{\partial \phi}}_{\text{calculate using Implicit Function theorem}}$$

$$\nabla_\phi \mathcal{L}_{\text{task}}(\theta^\star(\phi)) = -\nabla_\theta \mathcal{L}_{\text{task}}(\theta)\cdot\left(\frac{\partial \mathcal{L}_{\text{pred}}(\theta,\phi)}{\partial^2\theta}\right)^{-1}\cdot\frac{\partial \mathcal{L}_{\text{pred}}(\theta,\phi)}{\partial\phi\,\partial\theta}\bigg|_{\theta=\theta^\star(\phi)}$$

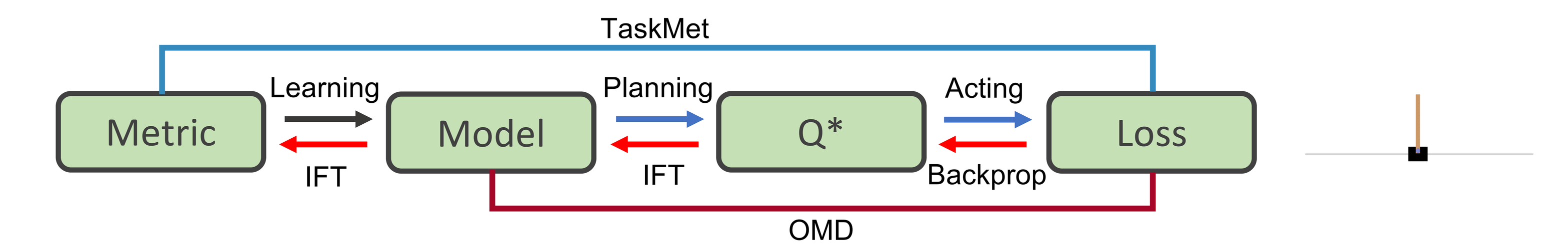
We approximately solve this with a **conjugate gradient** method



x: features
y: targets
Λ_ϕ: metric
$\mathcal{L}_{\text{pred}}(\theta,\phi) := \mathbb{E}\left[\|f_\theta(x)-y\|^2_{\Lambda_\phi(x)}\right]$

Experiments: model-based RL

Control-oriented model-based reinforcement learning with implicit differentiation. Nikishin et al., AAAI 2022.



Task: find the optimal Q value function

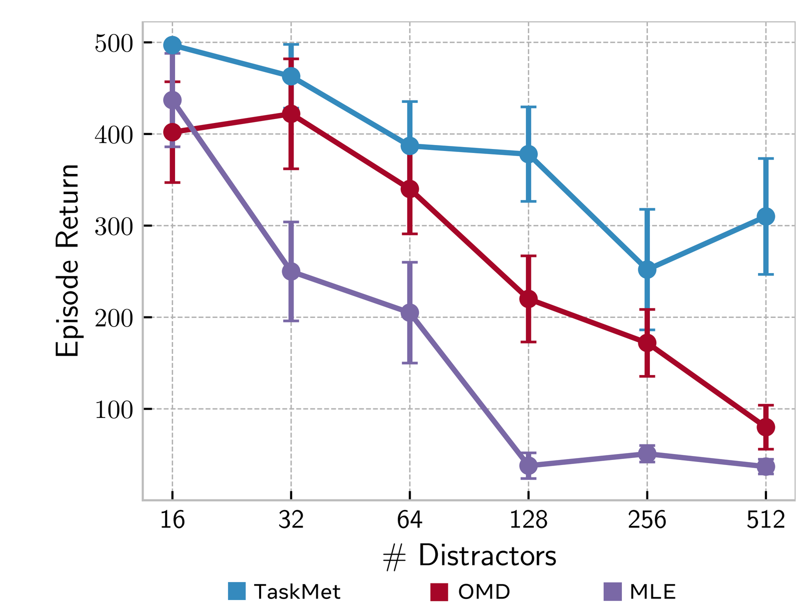
Environment: cartpole (#state dimensions: 4, #action dimensions: 2)

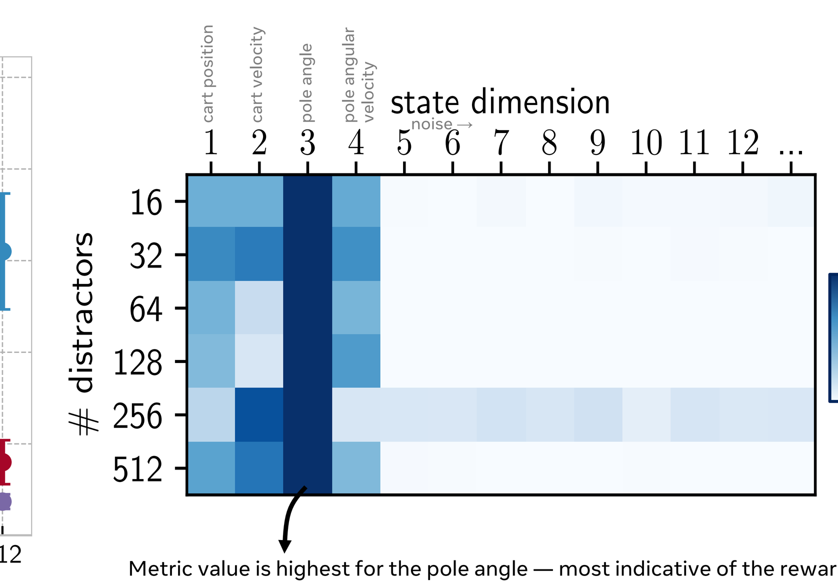
Get the **maximum return** using trajectories from **learned dynamics model**

Experiment : add noisy/distracting dimensions to the state space

Metric Λ: **diagonal** and **not conditioned** on x

Result: the learned metric downweights the noisy dimensions, allowing the prediction model to use its capacity on task relevant features only





Metric value is highest for the pole angle — most indicative of the reward

Experiments: decision-oriented model learning

Decision-Focused Learning without Differentiable Optimization. Shah et al., NeurIPS 2022.

Setup: model predictions parameterize a decision-making optimization problem

Example: portfolio optimization

$$\mathcal{L}_{\text{task}}(\theta) := \mathbb{E}_{x,y\sim D}[g(z^\star(\hat{y}_\theta(x)))]$$

decision quality; (negated) portfolio returns; portfolio allocation; stock features; predicted stock price; model

Cubic setting

Predict utilities with a linear model for a downstream maximization task

Severe modeling errors — must focus on high-utility points

Takeaway: our learned metric tilts the model to control the maximum prediction

normalized decision quality (0=random, 1=oracle)

Method	α (weight of MSE term)	Cubic	Budget	Portfolio
MSE		-0.96 ± 0.02	0.54 ± 0.17	0.33 ± 0.03
DFL	0	0.61 ± 0.74	0.91 ± 0.06	0.25 ± 0.02
DFL	10	0.62 ± 0.74	0.81 ± 0.11	0.34 ± 0.03
LODL	0	0.96 ± 0.005	0.84 ± 0.105	0.17 ± 0.05
LODL	10	-0.95 ± 0.005	0.58 ± 0.14	0.30 ± 0.03
TaskMet		0.96 ± 0.005	0.83 ± 0.12	0.33 ± 0.03