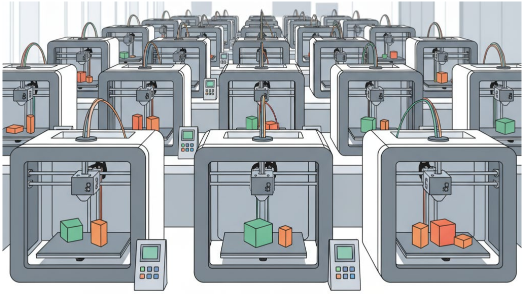
Speak to the Policy: Personalize MDP Policies with Language-based Priors

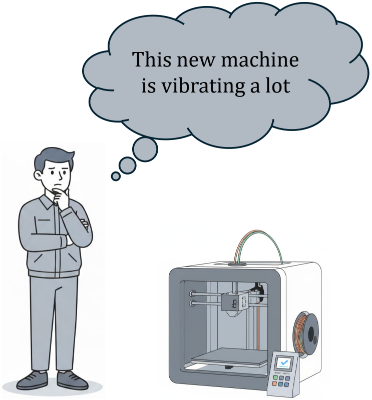
Qiyuan Chen and Raed Al Kontar

Industrial and Operations Engineering, University of Michigan, Ann Arbor

October 26, 2025

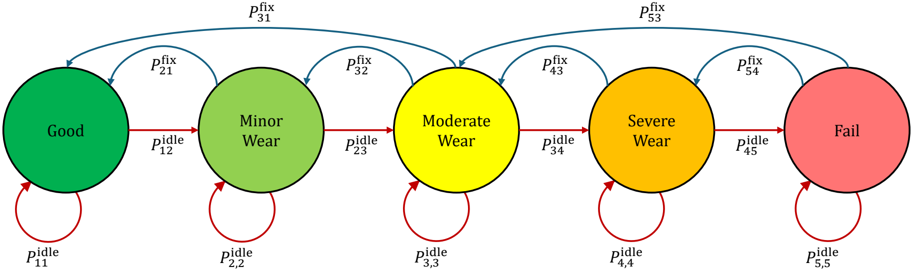
The Problem: How to "Speak" to Your Policy?



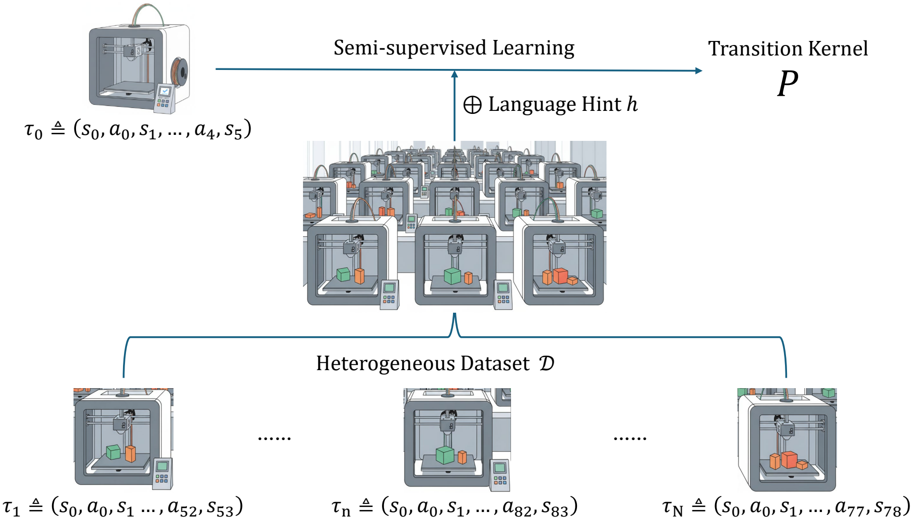


Background on MDP (An Example of Machine Maintenance)

- (Given) action $a \in \mathcal{A}$, state $s \in \mathcal{S}$, reward function: $R(s,a)$
- (To Be Learned) transition dynamics: $P_{ij}^{a} = \mathbb{P}(s_{t+1} = j \mid s_t = i, a)$



Problem Formulation: Data Flow



Problem Formulation: The Inputs

We are given three ingredients:

1. **Past Data**
   - The trajectory in the current environment: $\boldsymbol{\tau}_0$
   - n historical trajectories from other environments: $\mathcal{D} = \{\boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_n\}$
2. **Human Language Hint**
   - A text hint h describing the current environment behavior
3. **A Language Model**
   - An LLM that can understand h

Goal

Learn a decision policy for the current environment with the help of the hint h the dataset $\mathcal{D}$

Overall Approach: A Two-Stage Pipeline

Our algorithm consists of two main stages:

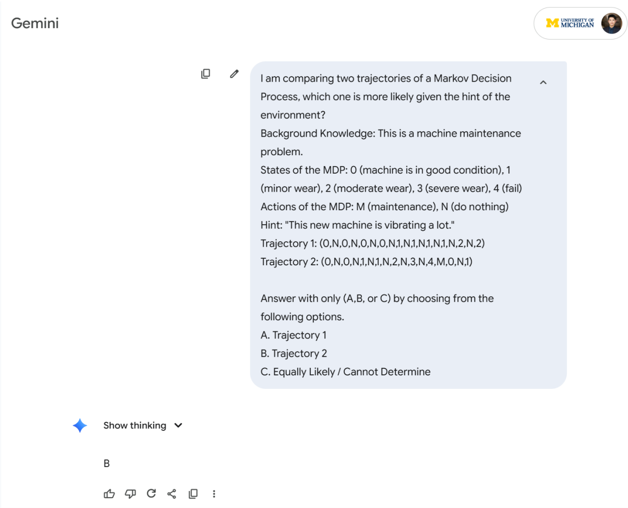
Stage 1: Generate Priors via LLM

- Use the LLM and hint h to “judge” how likely trajectories lies in the main component
- Train a model to produce a prior score s_k for each trajectory $\boldsymbol{\tau}_k$

Stage 2: Robust Kernel Estimation

- Use a semi-supervised, prior-weighted EM algorithm
- This algorithm combines the labeled data $\boldsymbol{\tau}_0$ and the unlabeled data $\mathcal{D}$
- The priors generated in Stage 1 guide the EM algorithm to filter out outliers

Step 1.1: LLM as a Judge



Collecting LLM's judgments

We input the following to the LLM:

- Background Knowledge
- Trajectory 1: τ_i
- Trajectory 2: τ_j
- Hint: h
- Prompt

The LLM outputs a preference:

- A. τ_i is more likely
- B. τ_j is more likely
- C. Cannot Determine

Step 1.1: LLM as a Judge (cont.)

Gemini

Hint: "This new machine is vibrating a lot."

Trajectory 1: (0,N,0,N,0,N,0,N,1,N,1,N,1,N,1,N,2,N,2)

Trajectory 2: (0,N,0,N,1,N,1,N,2,N,3,N,4,M,0,N,1)

Show thinking

B

Stage 1: Generating Language-based Priors

Stage 1: Generate Priors via LLM

- Step 1.1 collects a dataset of M pairwise preferences, $M = \mathcal{O}(n \log_2 n)$
- Step 1.2 trains a scoring function $f_\theta(\boldsymbol{\tau})$ that maps any trajectory $\boldsymbol{\tau}$ to a scalar score

Bradley-Terry Model

This model finds scores $f_\theta(\boldsymbol{\tau})$ such that the probability of $\boldsymbol{\tau}_i$ being preferred to $\boldsymbol{\tau}_j$ is a logistic function of their score difference:

$$\mathbb{P}(\boldsymbol{\tau}_i \succ \boldsymbol{\tau}_j) = \sigma(f_\theta(\boldsymbol{\tau}_i) - f_\theta(\boldsymbol{\tau}_j))$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the standard sigmoid function.

Step 1.2: Optimizing the Trajectory Score Model

We minimize the negative log-likelihood (given by B-T Model) of the M observed preferences:

$$\min_{\theta} \quad -\frac{1}{M} \sum_{(\boldsymbol{\tau}_i \succ \boldsymbol{\tau}_j)} \log \frac{\exp(f_\theta(\boldsymbol{\tau}_i))}{\exp(f_\theta(\boldsymbol{\tau}_i)) + \exp(f_\theta(\boldsymbol{\tau}_j))}$$

Result: Prior Scores

After training, we can compute a score for every unlabeled trajectory:

$$f_\theta(\boldsymbol{\tau}_k), \quad \forall k \in \{1, \ldots, n\}$$

This s_k is our prior belief that $\boldsymbol{\tau}_k$ belongs to the main component.

Stage 2: Model Assumptions

We use a semi-supervised Expectation-Maximization (EM) algorithm.

Mixture Model

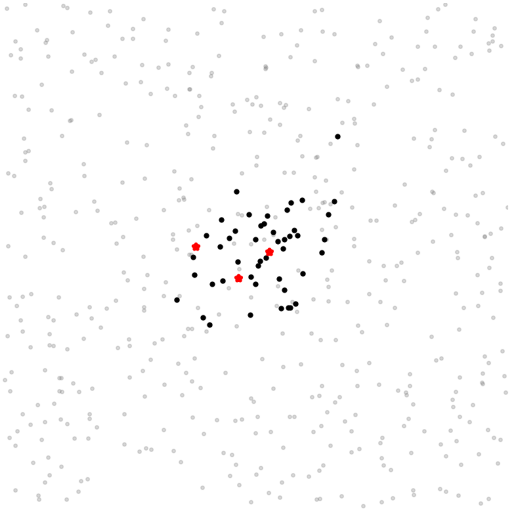
We assume each trajectory $\boldsymbol{\tau}_k$ is drawn from a two-component mixture, determined by a latent variable $c_k \in \{\text{main}, \text{outlier}\}$.

- If $c_k = \text{main}$: $\boldsymbol{\tau}_k$ is generated from P (our goal).
- If $c_k = \text{outlier}$: $\boldsymbol{\tau}_k$ is generated from a fixed uniform noise component.

Semi-Supervised

- For labeled data $\boldsymbol{\tau}_0$, we fix $c_0 = \text{main}$.
- For unlabeled data $\boldsymbol{\tau}_k$, c_k is a latent variable.

Model Assumptions (Visual)



Warning: Oversimplification for Intuitions

Components:

- Main component (black)
- Observed samples (red)
- Background noise (gray)

Correspondence:

- τ_0: Red
- $\mathcal{D}$: Black+ Gray

Stage 2: EM Algorithm - Initialization

Step 1: Calculate Priors (for $\boldsymbol{\tau}_k \in \mathcal{D}$)

Convert scores to prior probabilities:

$$\pi_k \triangleq \mathbb{P}(c_k = \text{main}) \approx \sigma(f_\theta(\boldsymbol{\tau}_k) - \kappa),$$

where κ is a hyperparameter that controls the background noise level.

* A simple choice of κ can be $\kappa = \frac{1}{n}\sum_k f_\theta(\boldsymbol{\tau}_k)$.

Step 2: Initialize Main Kernel P

Set initial estimate using *only* the labeled data $\boldsymbol{\tau}_0$ (a is suppressed):

$$P_{ij} = \frac{N_{ij}^k}{\sum_{l=1}^{S} N_{il}^k}$$

Stage 2: EM Algorithm - The Loop

We repeat the E-Step and M-Step until the kernel P converges.

E-Step (Expectation)

Calculate the posterior probability (responsibility) r_k that each trajectory $\boldsymbol{\tau}_k$ belongs to the main component, given the current kernel P.

M-Step (Maximization)

Update the kernel P by calculating the weighted MLE, using the responsibilities r_k as weights.

Stage 2: EM Algorithm - E-Step

For Labeled Data ($\boldsymbol{\tau}_0$)

The responsibility $r_0 = \mathbb{P}(c_0 = \text{main} | \boldsymbol{\tau}_0, P)$ is fixed to our certain belief: $r_0 = 1$

For Unlabeled Data ($\boldsymbol{\tau}_k \in \mathcal{D}$)

Find the posterior responsibility using Bayes' rule:

$$r_k = \mathbb{P}(c_k = \text{main} | \boldsymbol{\tau}_k, P) = \frac{\overbrace{\pi_k}^{\text{Langauge Prior}} \cdot \overbrace{\mathbb{P}(\boldsymbol{\tau}_k | c_k = \text{main}, P)}^{\text{Likelihood (Main)}}}{\pi_k \cdot \mathbb{P}(\boldsymbol{\tau}_k | c_k = \text{main}, P) + (1 - \pi_k) \cdot \mathbb{P}(\boldsymbol{\tau}_k | c_k = \text{noise}, P)}$$

$$\mathbb{P}(\boldsymbol{\tau}_k | c_k = \text{main}, P) = \prod_{i=1}^{S} \prod_{j=1}^{S} (P_{ij})^{N_{ij}^{(k)}}, \quad \mathbb{P}(\boldsymbol{\tau}_k | c_k = \text{noise}, P) = \left(\frac{1}{S}\right)^{|\boldsymbol{\tau}_k|}$$

Stage 2: EM Algorithm - M-Step

We update P by calculating the weighted MLE

- "Hard" counts from labeled data $\boldsymbol{\tau}_0$
- "Soft" (responsibility-weighted) counts from unlabeled data $\mathcal{D}$

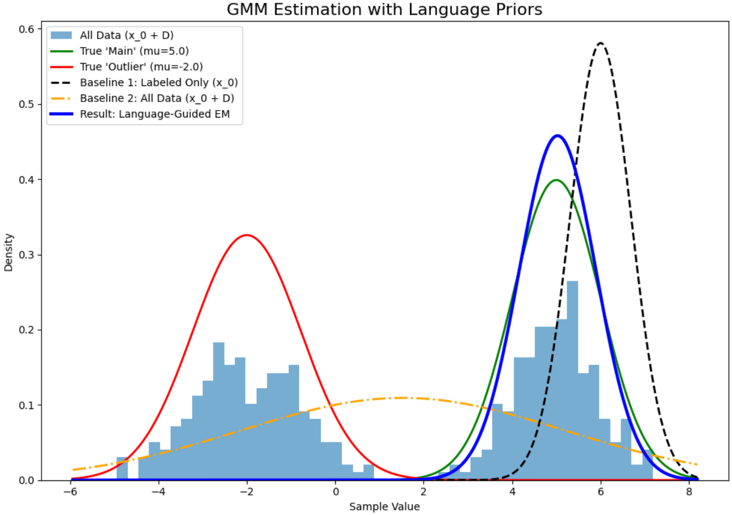
Update Rule

For each transition pair $i \to j$:

$$P_{ij} = \frac{N_{ij}^{(0)} + \left(\sum_{k \in \mathcal{D}} r_k \cdot N_{ij}^{(k)}\right)}{\sum_{l=1}^{S} \left[N_{il}^{(0)} + \left(\sum_{k \in \mathcal{D}} r_k \cdot N_{il}^{(k)}\right)\right]}$$

Repeat E-Step and M-Step until P converges.

Visualizations on a Gaussian Mixture Model



Toy Example:

- Demo with 1-d GMM
- Hint: “values are larger”
- $\kappa = \frac{1}{n} \sum_k f_\theta(\boldsymbol{\tau}_k)$

Highlights:

- Beyond MDP
- Robust Recovery
- Non-uniform noise

Thank You for Listening!

Speak to the Policy

- **Stage 1** An LLM judges trajectories based on a human instruction h. A choice model is trained on these judgments to create prior scores.
- **Stage 2** The language prior guides the semi-supervised learning of the transition kernels. The model is optimized by an EM algorithm.

Qiyuan-Chen.com