machine learning conference

The Recurrent Recommender Bandit: Neural Recommenders, Optimized for Engagement

Ben Allison baalliso@
ML Scientist | Amazon Development Centre Scotland

A New Way to Think of Recommendations

- Existing ML recommenders try to predict behaviour...
  - ...and recommend the most likely products
- There are known cases where this doesn't work:
  - Popularity bias (Harry Potter effect)
  - Products that don't work for recommendations (e.g. Batteries)
  - Predicting what customers were going to do anyway
- How can we optimize what to recommend automatically?

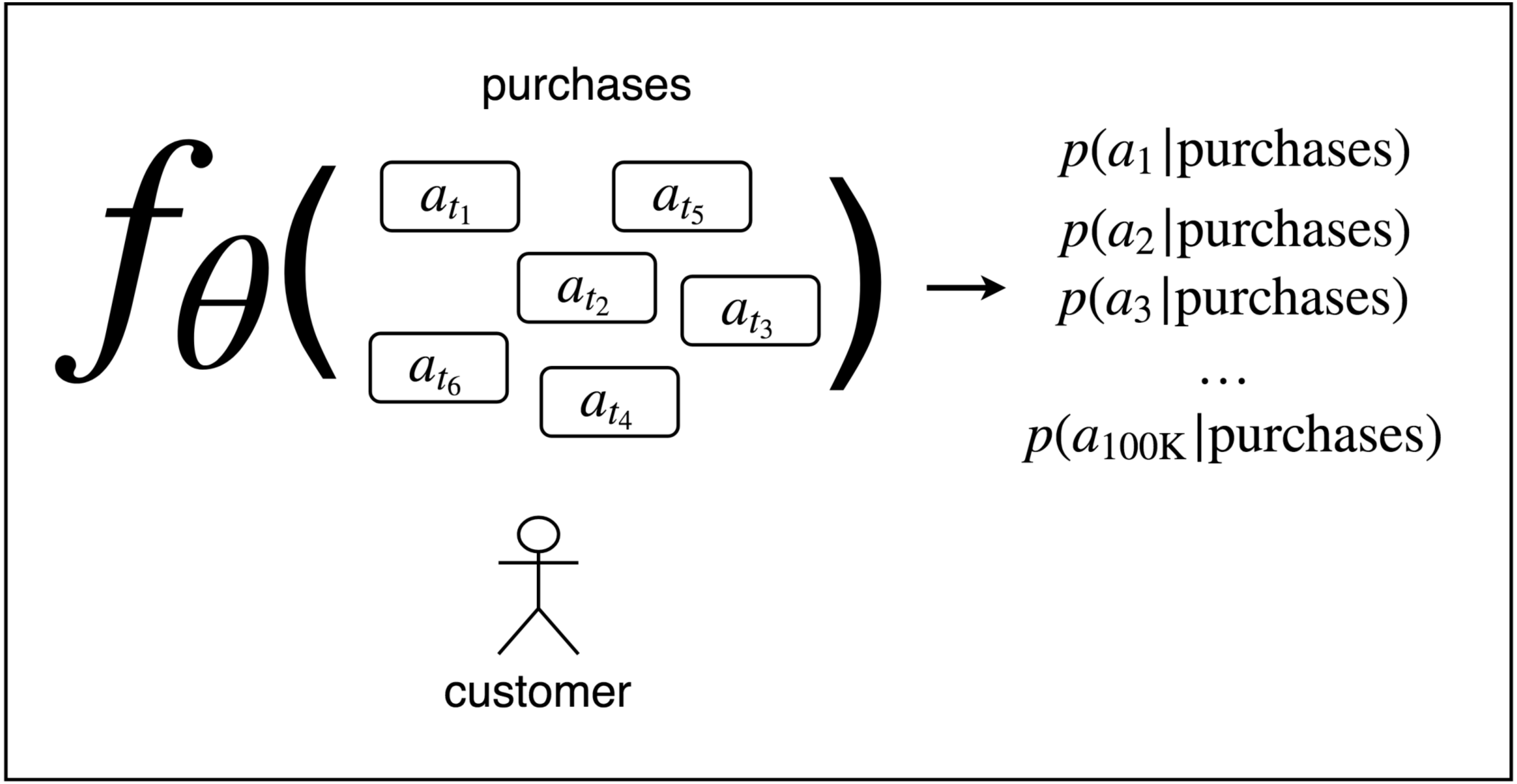
Application

- General-purpose recommendations technology
- Today:
  - Purchase-based product recommendations
  - Multiple locations (main page, recent history footer)
  - 20 categories
- Imminent expansion to other venues/contexts/problems

Our Approach

- Recommendations is a partial feedback problem
  - Solve with bandit/reinforcement learning
- What we recommend determined by recommendations *policy*
- Useful input to this policy: good predictive model
- Improve recommendations by optimizing policy automatically

Parametric Recommenders



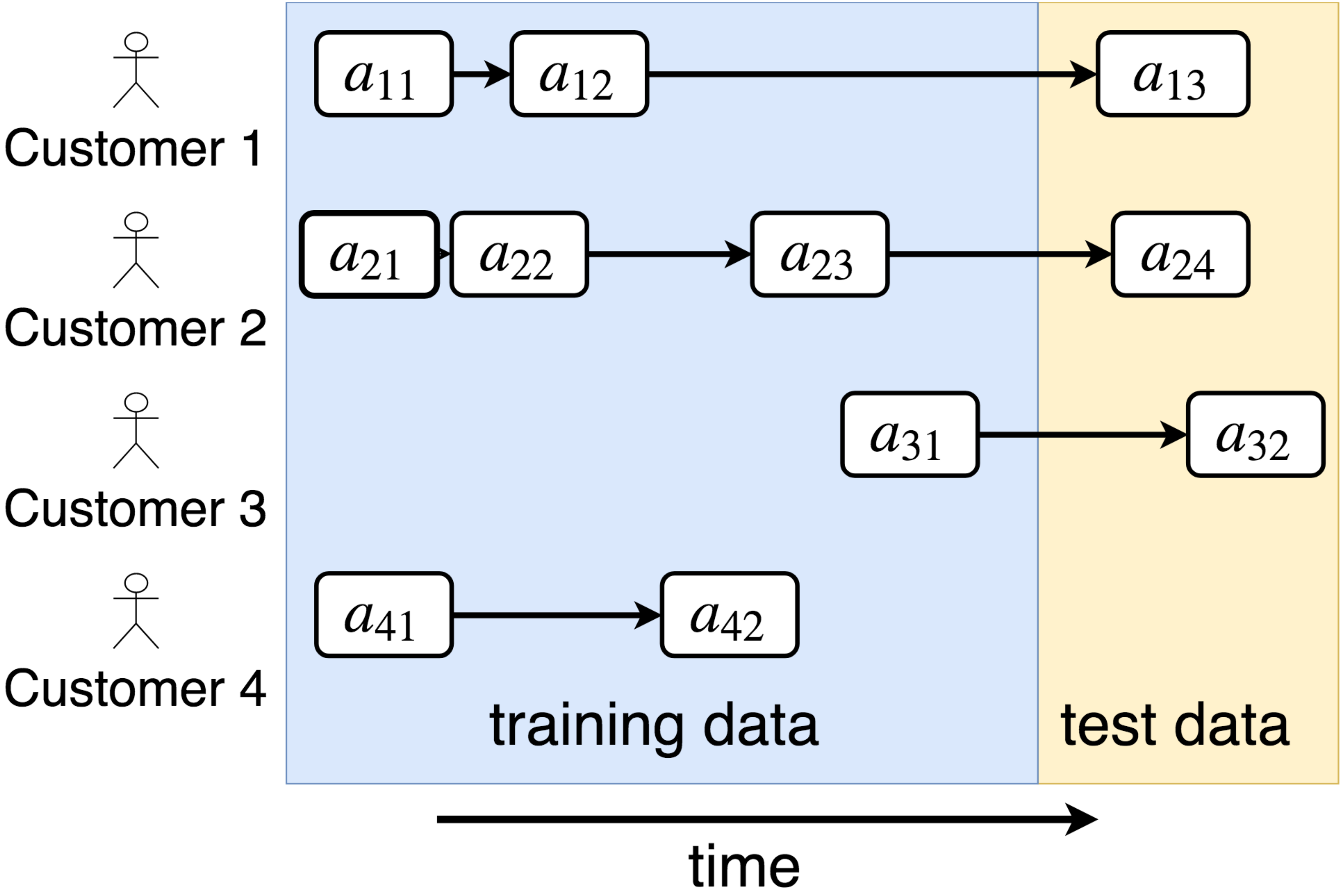
Parametric Recommenders

purchases

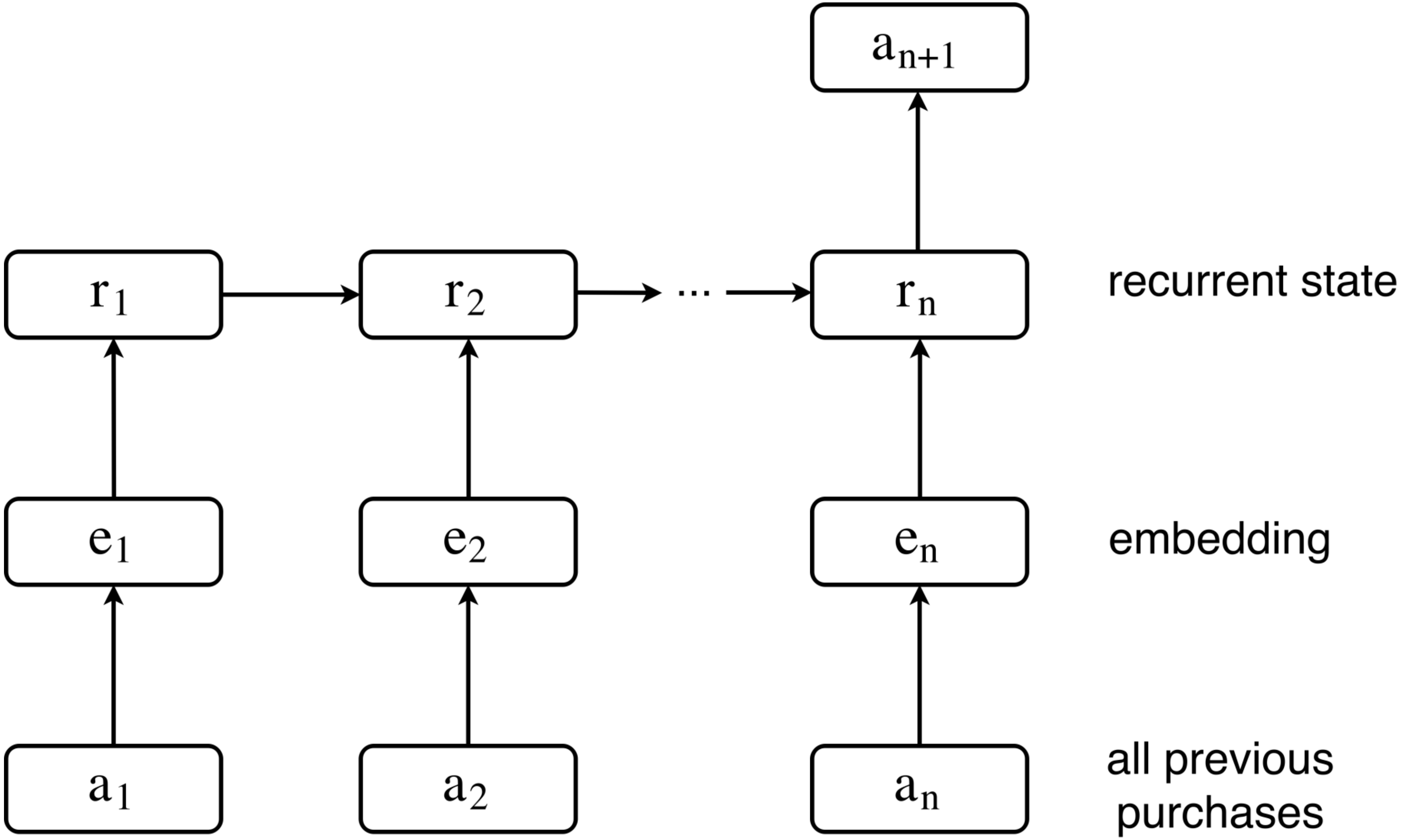
$$f_\theta\left(\boxed{a_{t_1}}\ \boxed{a_{t_5}}\ \boxed{a_{t_2}}\ \boxed{a_{t_3}}\ \boxed{a_{t_6}}\ \boxed{a_{t_4}}\right) \rightarrow \begin{array}{c} p(a_1 \mid \text{purchases}) \\ p(a_2 \mid \text{purchases}) \\ p(a_3 \mid \text{purchases}) \\ \cdots \\ p(a_{100\text{K}} \mid \text{purchases}) \end{array}$$

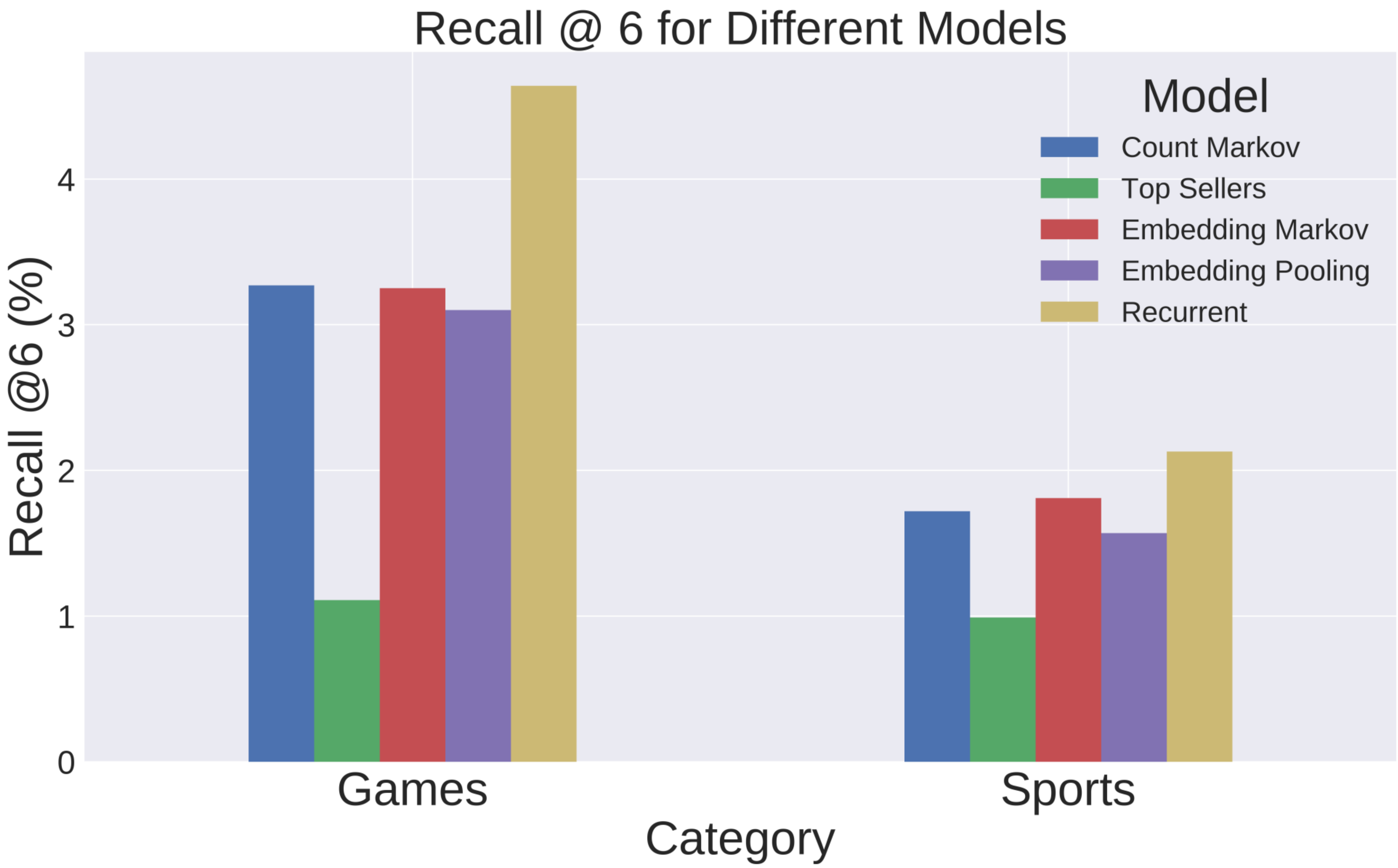
$$\theta^\star = \arg\min_\theta \mathcal{L}(\theta)$$

Experimental Setup

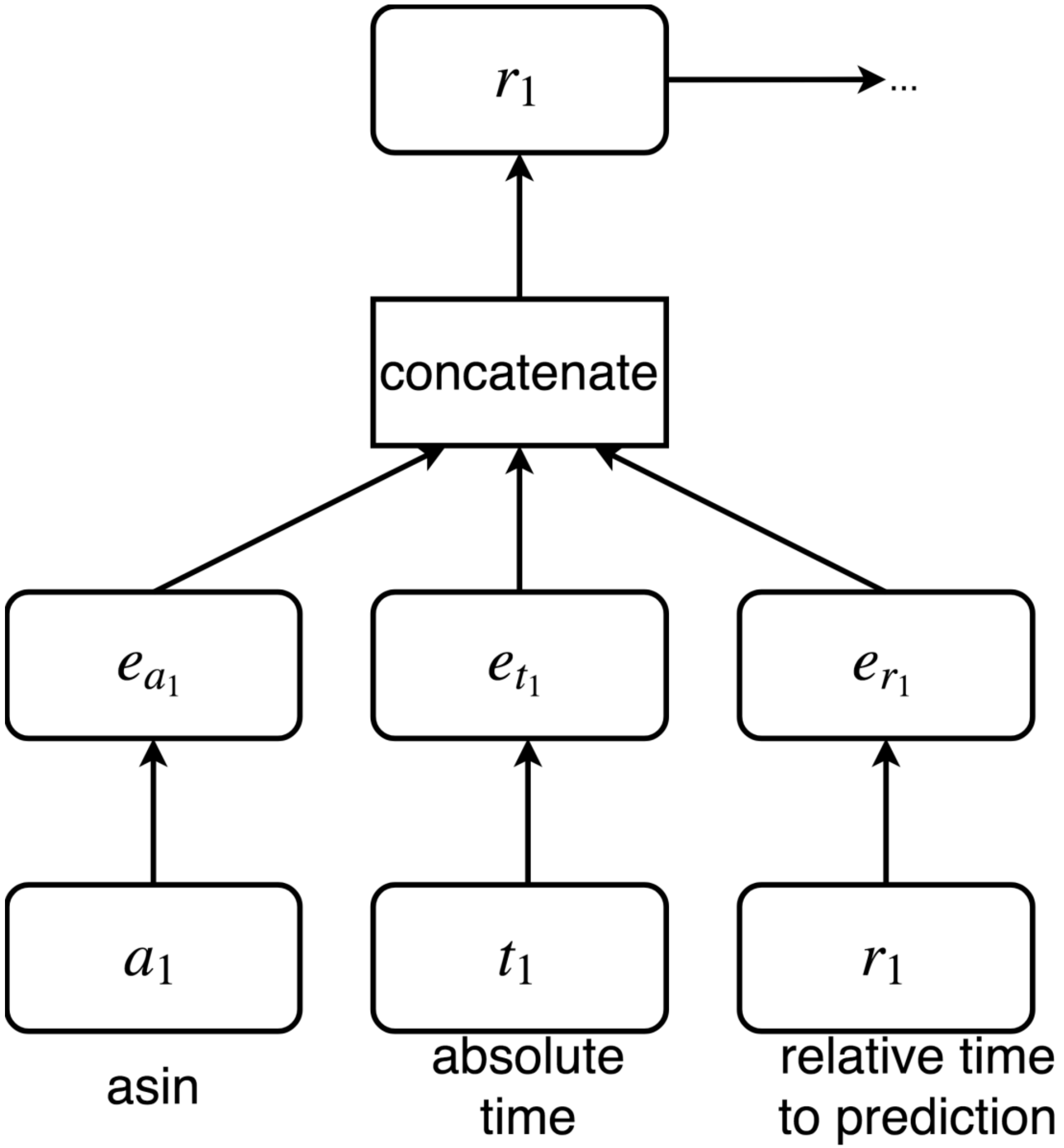


Predicting Purchases with an RNN

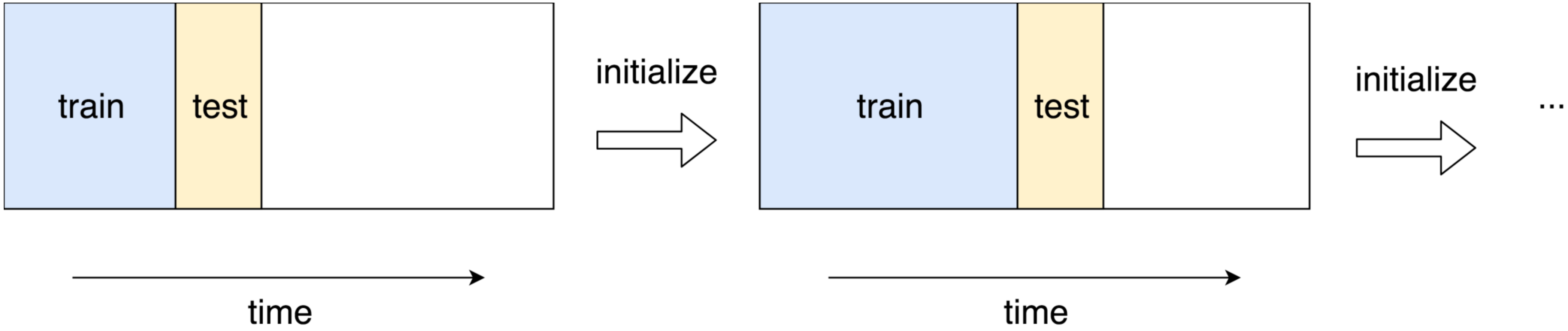


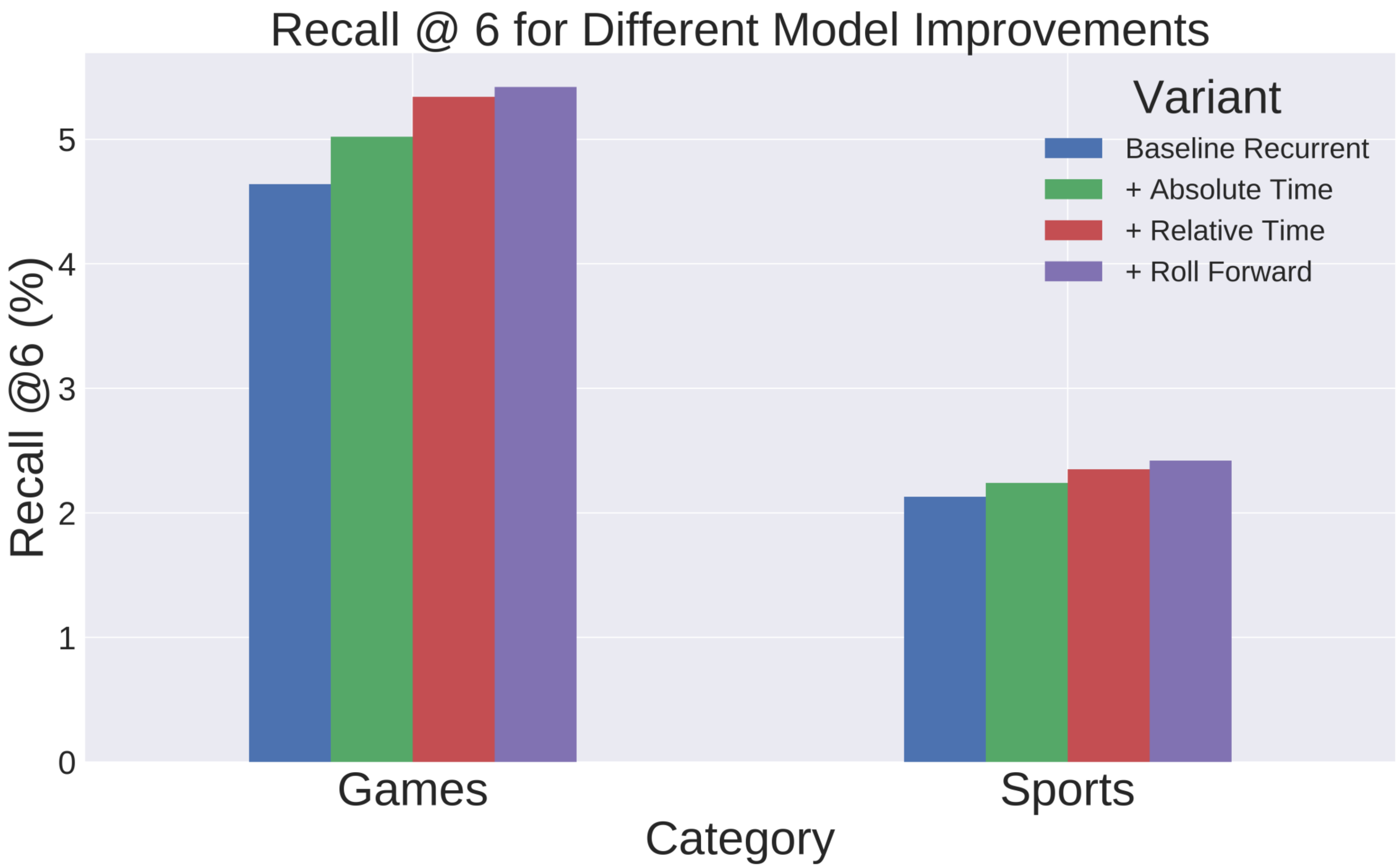

Recall @ 6 for Different Models
Model
Count Markov
Top Sellers
Embedding Markov
Embedding Pooling
Recurrent
Recall @6 (%)
0
1
2
3
4
Games
Sports
Category

Policy Improvement: Time Information



Policy Improvement: Roll-Forward Training




Recall @ 6 for Different Model Improvements
Variant
Baseline Recurrent
+ Absolute Time
+ Relative Time
+ Roll Forward
Recall @6 (%)
0
1
2
3
4
5
Games
Sports
Category

Incorporating Feedback: Policy Optimization

- Recommend by sampling from predictive distribution
- Sometimes works really well:
  - Can we do this more often?
- Sometimes doesn't work:
  - Can we stop doing this?
- Solution: use what we show, and outcomes, to get better!

Engagement Loss

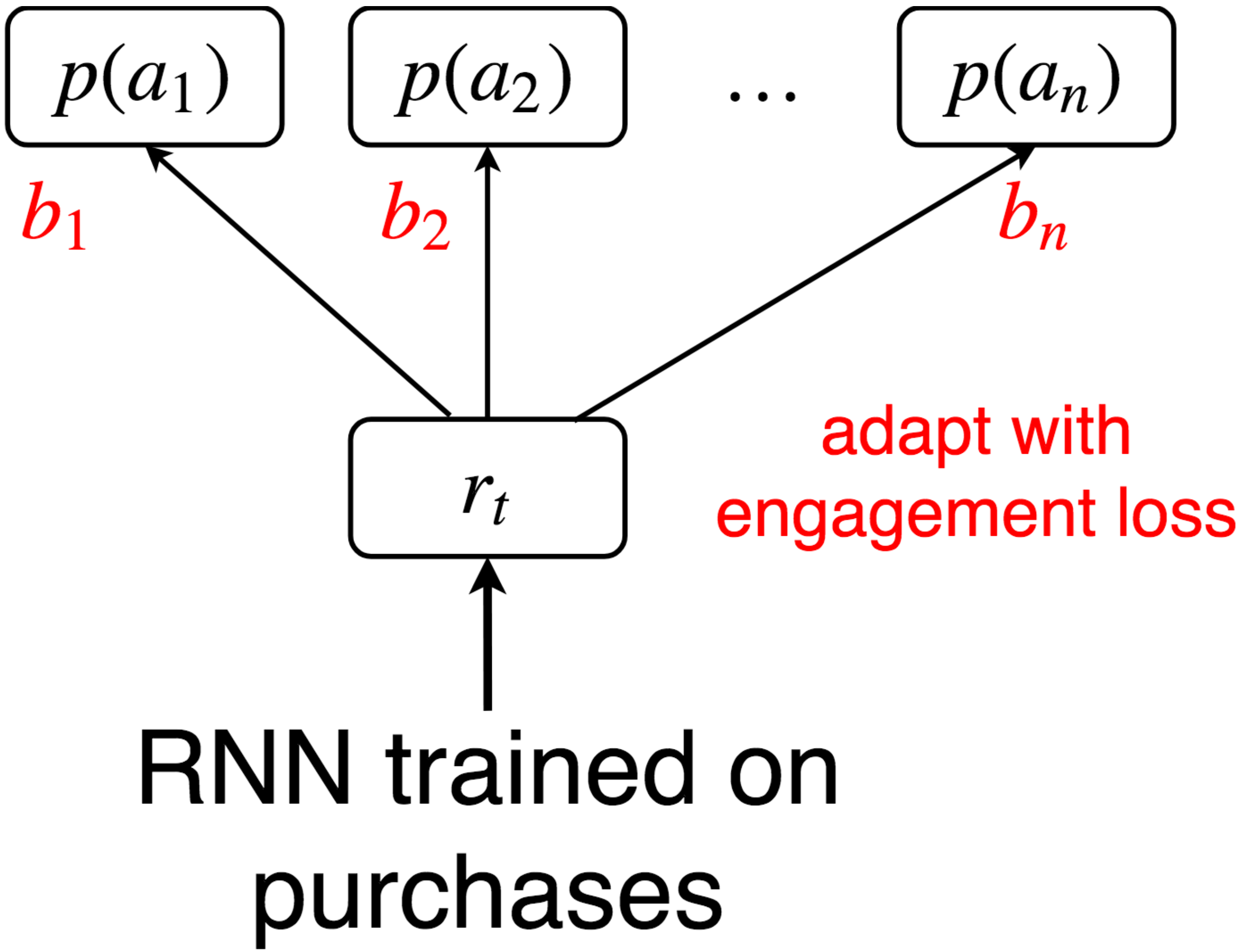
$$\mathcal{L}(\theta) = \mathbb{E}_{p(\text{customer})} \frac{p_\theta(a|\text{customer})}{p'(a|\text{customer})} \mathbb{1}_{\text{click}}$$

- Sum over logged data
- Logged probability
- Outcome (did customer click?)
- Our RNN

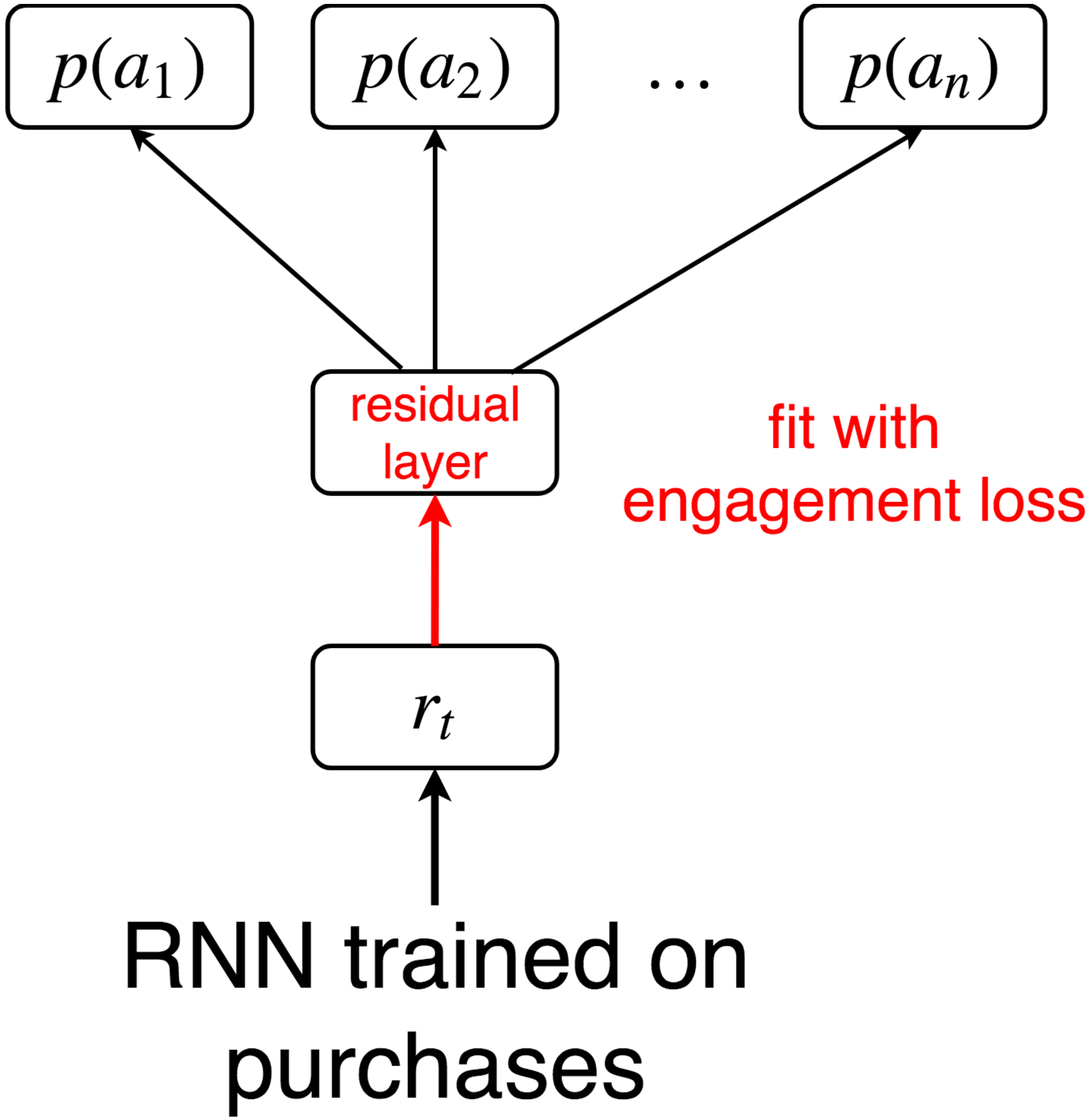
Engagement Loss as Adaptation

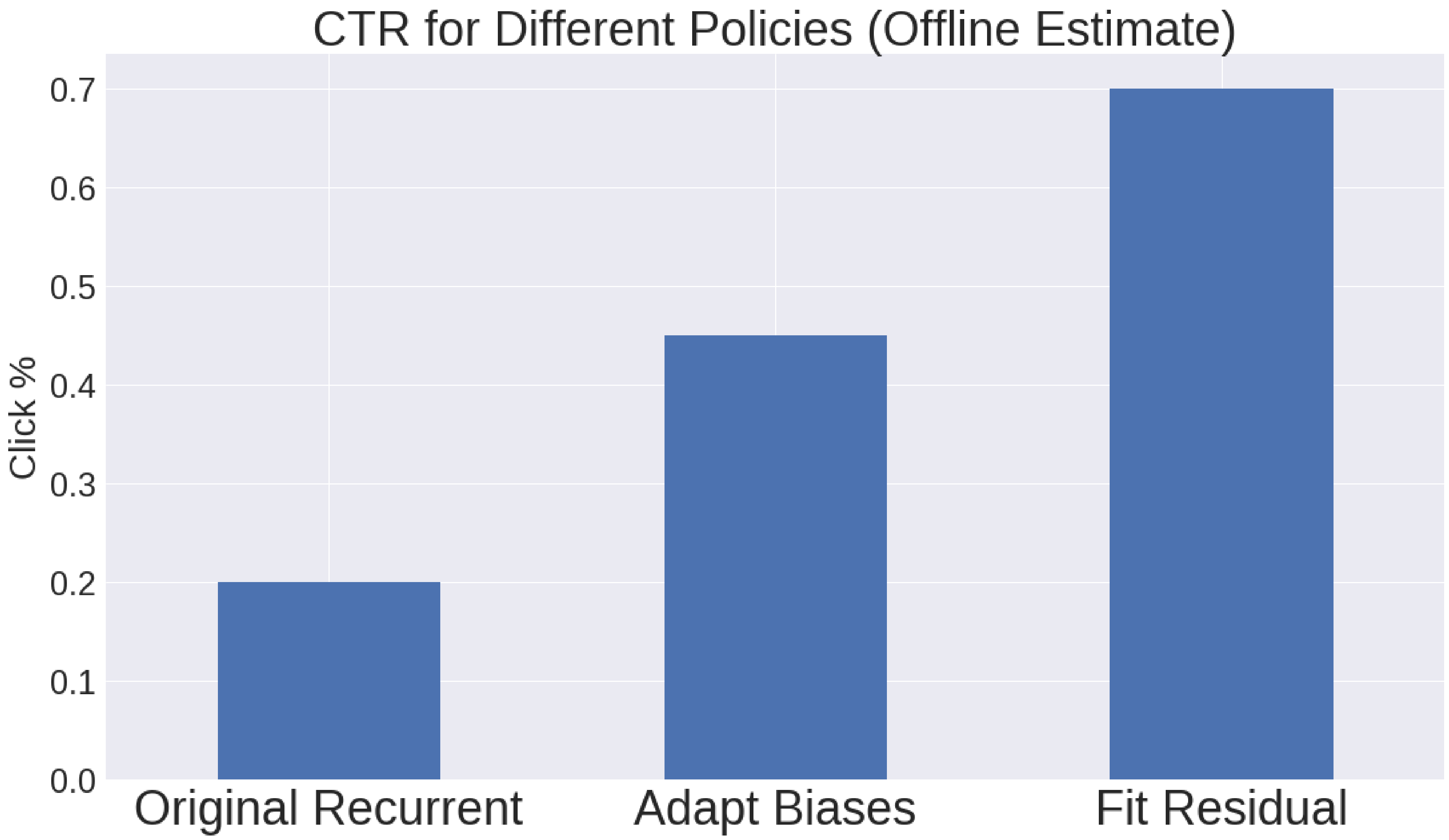
- Cannot train policy using engagement loss alone
- First train to predict purchases
- Learn adaptation with engagement loss

Feedback Loss: Adapt Biases



Feedback Loss: Fit Residual




CTR for Different Policies (Offline Estimate)
Click %
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Original Recurrent
Adapt Biases
Fit Residual

Summary

- New general-purpose way of approaching recommendations
- Acknowledge recommendations is not a prediction problem
- Explore powerful policies for making recommendations
- Refine these policies using logged feedback data

Collaborators

- Co-Authors: Roberto Pellegrini, Dave Turner, Iain Murray
- Engineering Support: Personalization in Edinburgh