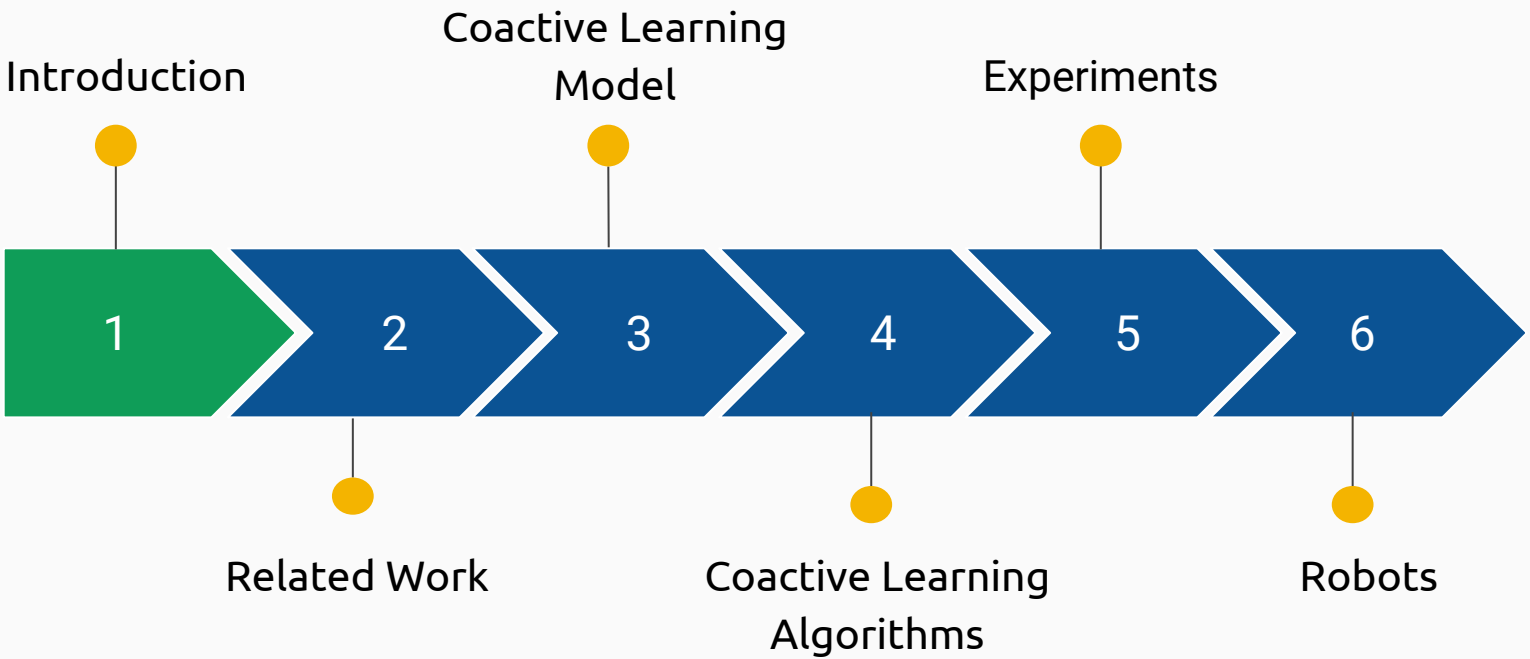
Coactive Learning

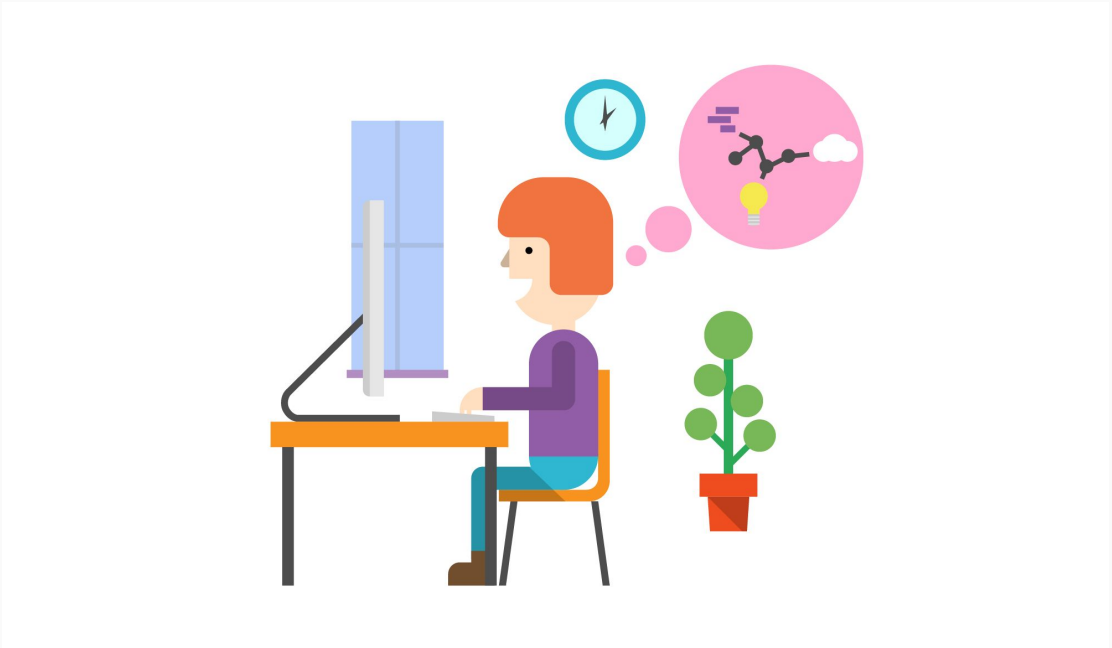
Rohan Batra, Avishek Dutta, Nand Kishore, Siddharth Murching

Roadmap


Introduction
Coactive Learning Model
Experiments
1
2
3
4
5
6
Related Work
Coactive Learning Algorithms
Robots

Introduction

- Interaction between humans and most systems today:
  - User issues a command
  - User receives a result
  - User interacts with the results
- In this way, the user provides implicit feedback about his/her utility function



Web Search

- User types the query ‘Who is Yisong Yue?’
- Search engine presents the ranking to the right
- User clicks on documents B and D

A

Yisong Yue | Machine Learning Researcher @ Caltech
www.yisongyue.com/
Yisong Yue is an assistant professor in the Computing and Mathematical Sciences Department at the California Institute of Technology. His research interests lie ...

- **Research**
  His research interests lie primarily in the theory and application of ...
- **About**
  Yisong Yue is an assistant professor in the Computing and ...
- **Please read before emailing**
  (please read before emailing) ... please clearly state WHAT ...
- **CS/CNS/EE 155: Machine ...**
  (CS/CNS/EE 155) Machine ... in machine learning and data ...

More results from yisongyue.com »

B

Yisong Yue | LinkedIn
https://www.linkedin.com/in/yisongyue
Pasadena, California - Assistant Professor at Caltech - Caltech
Join LinkedIn and access **Yisong's** full profile. ... Research in theory and application of statistical machine learning. ... Bachelor of Science, Computer Science.

C

Random Ponderings
yyue.blogspot.com/
Jan 1, 2016 - by Taehwan Kim, **Yisong Yue**, Sarah Taylor, and Iain Matthews I'll start with a shameless piece of self-advertising. In collaboration with Disney ...

D

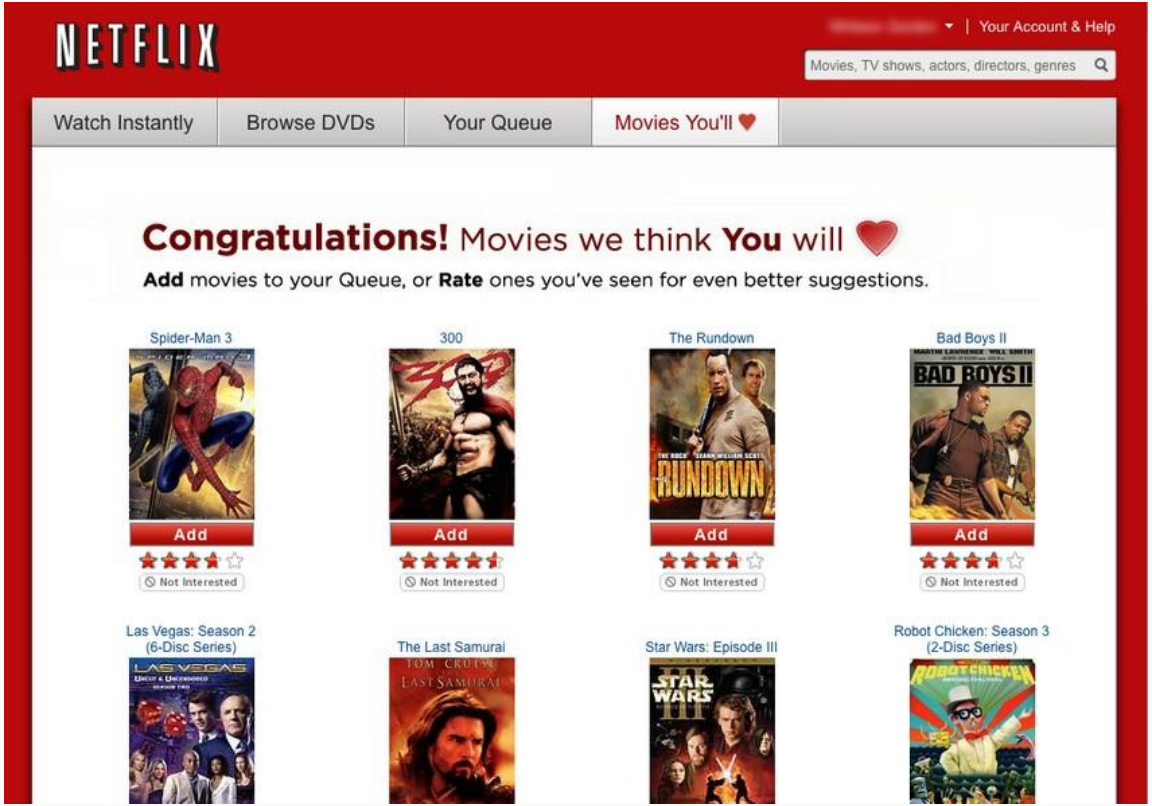
Yisong Yue - Google Scholar Citations
https://scholar.google.com/citations?user=tEk4qo8AAAAJ Google Scholar
California Institute of Technology - caltech.edu
Interactively optimizing information retrieval systems as a dueling bandits problem. Y **Yue**, T Joachims. Proceedings of the 26th Annual International Conference ...

E

Yisong Yue - Computing + Mathematical Sciences - Caltech
www.cms.caltech.edu/people/5388/profile California Institute of Technology
Research Overview. **Yisong Yue's** research interests lie primarily in the theory and application of statistical machine learning. He is particularly interested in ...

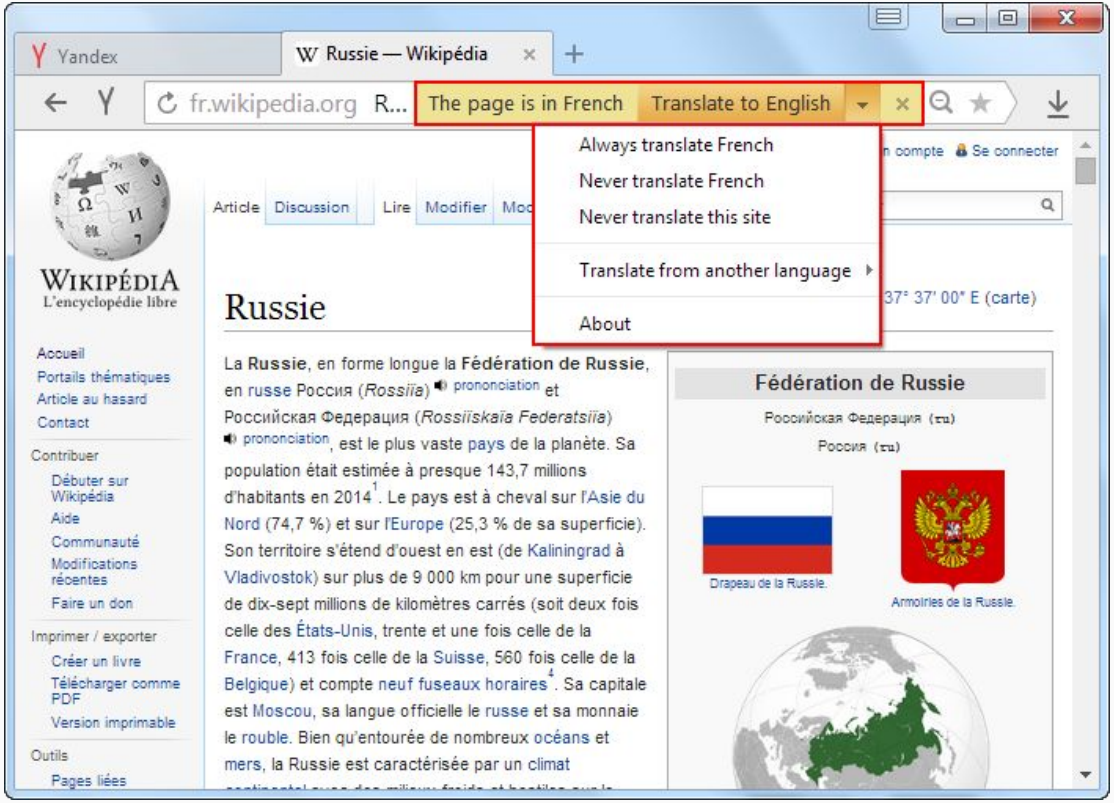
Movie Recommendation

- User watches a number of movies
- Netflix makes recommendations
- User rents movie D after viewing all options



Machine Translation

- User requests an online machine translator to translate a wiki page from language A to B
- System returns translated page
- User manually corrects some of the translated text



Implicit Feedback

- In all the previous examples, the user provides implicit feedback

- Typically, this feedback is only an incremental improvement
  - Web page B is better than web page A

- Often very difficult for a human to specify the optimal result/prediction
  - Web pages should be ranked B,D,C,A,E

Key Contributions

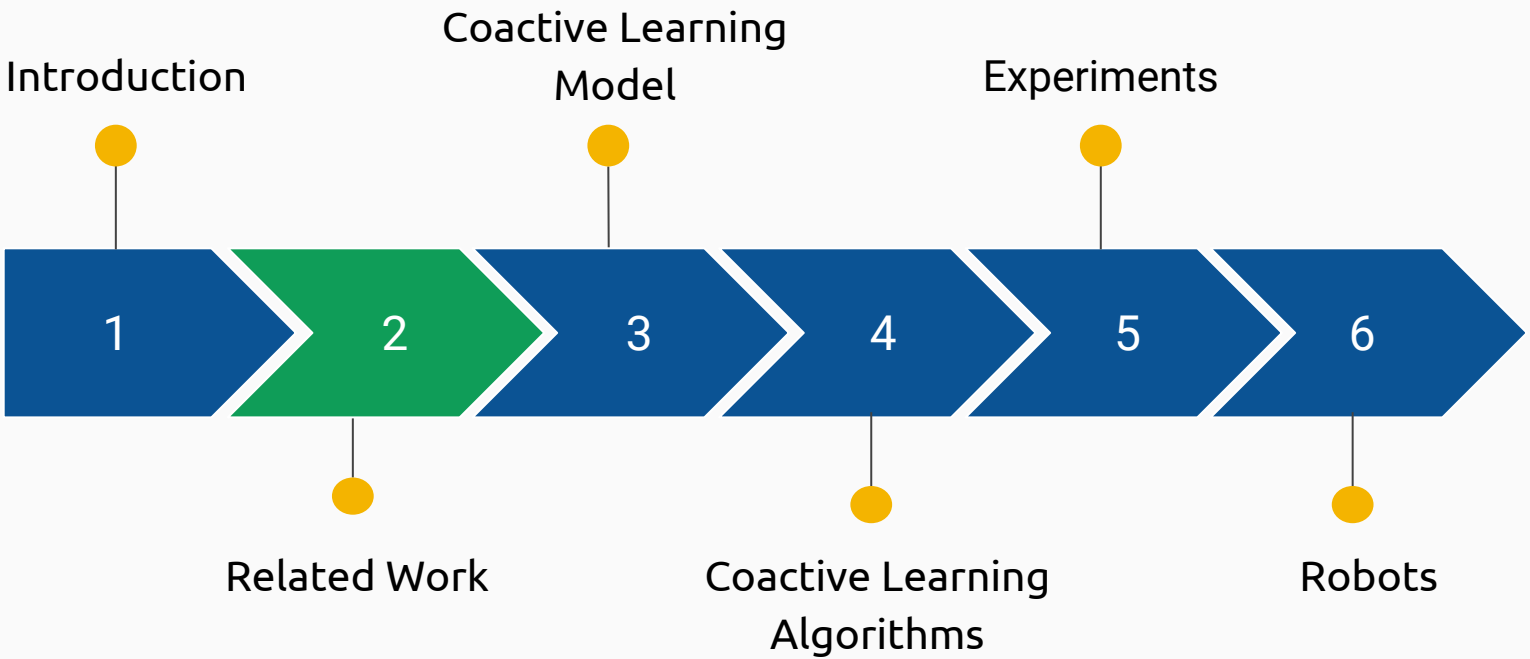
1. Formalize Coactive Learning as a model of interaction between a learning system and its user
   - Define notion of regret
   - Validate assumption of implicit user feedback

2. Derive learning algorithms for the Coactive Learning Model
   - Linear utility models
   - Convex cost functions
   - Show $\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$ regret bounds

3. Provide empirical evaluations of algorithms
   - Web search

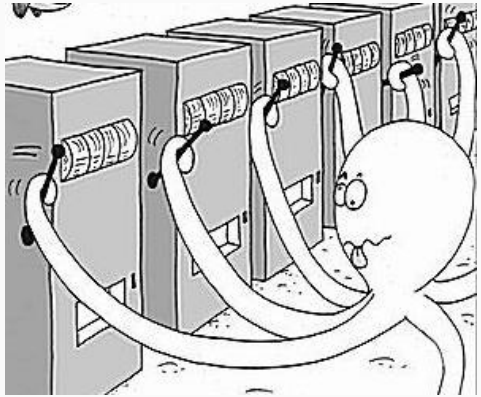
4. Robotic Application

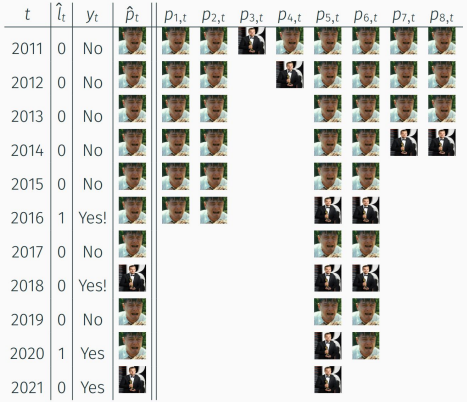
Roadmap


Introduction
Coactive Learning Model
Experiments
1
2
3
4
5
6
Related Work
Coactive Learning Algorithms
Robots

Related Work

- The Coactive Learning Model bridges the gap between two previously studied forms of feedback
  - Multi-armed Bandit Model
    - We choose one arm and learn the utility of that arm only
  - Learning with Experts Model
    - We choose one arm, but learn the utility of all arms
- Our model reveals information about two arms at every timestep





Related Work

- Both the aforementioned models can be relaxed to the continuous setting
  - Multi-armed Bandit Model
    - Online Convex Optimization in the Bandit Setting
  - Learning with Experts Model
    - Online Convex Optimization
- Our model explores the continuous case with the Convex Preference Perceptron Algorithm

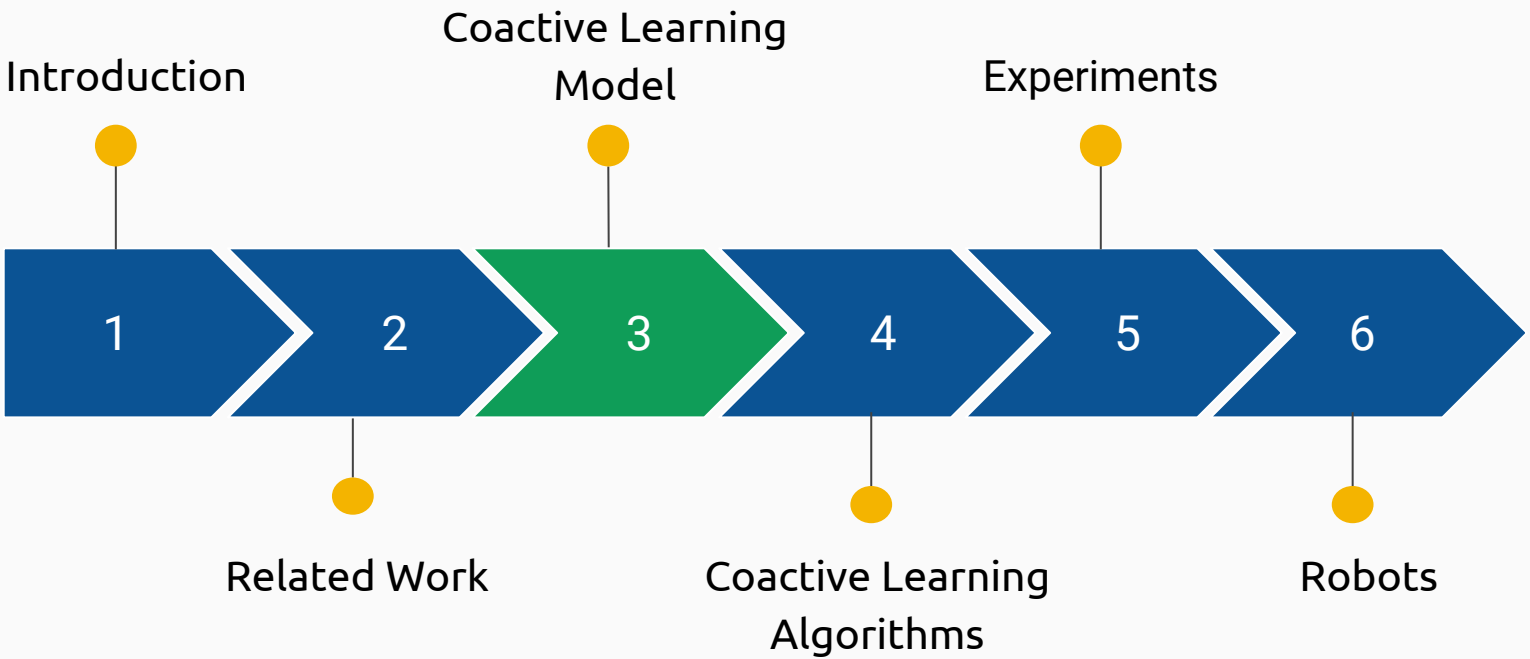
Related Work

- The most closely related problem is the Dueling Bandits Problem
- Recall that in this setting, the learning algorithm presents two arms to the user in some interleaved format
- The user gives implicit feedback that can be used to construct a pairwise ordering of the arms
- Our setting is **different** in that only one arm is given to the user
- The other is implicitly determined from user feedback
  - More details about this when we explain the model and experimental results

Comparison of Similar Frameworks

Framework	Algorithm	Feedback
Bandits	pull an arm	observe cardinal reward for the arm pulled
Experts	pull an arm	observe cardinal rewards for all the arms
Dueling Bandits	pull two arms	observe feedback on which one is better
Coactive Learning	pull an arm	observe another arm which is better

Roadmap


Introduction
Coactive Learning Model
Experiments
1
2
3
4
5
6
Related Work
Coactive Learning Algorithms
Robots

Coactive Learning Model

- In the coactive learning model, both the human and the learning algorithm have the same goal of obtaining good results

- In each round t
  - Learning algorithm observes context $\mathbf{x_t} \in \mathcal{X}$
  - Algorithm presents a structured object 
  - User “returns” an improved object $\bar{\mathbf{y}}_\mathbf{t} \in \mathcal{Y}$

Coactive Learning Model - Utility

- The utility of y_t within the context x_t is given as the unknown function $U(\mathbf{x_t}, \mathbf{y_t})$

- Generally, the learned feedback, $\bar{y}_t$ satisfies:

$$U(x, \bar{y}_t) > U(x, y_t)$$

- We will also allow violations of this condition through α-informative user feedback

Coactive Learning Model - Regret

- Our algorithm should return objects with utility close to that of the optimal
- Thus, if our algorithm presents object y_t under context x_t at time t, it suffers a regret:

$$\sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t)$$

- The average regret over T steps is:

$$REG_T = \frac{1}{T} \sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t)$$

- Goal of the algorithm is to minimize this average regret

Coactive Learning Model - User Feedback

- User generates feedback $\bar{y}_t$ through an approximate utility-maximizing search over a subset $\bar{\mathcal{Y}}_t$ of $\mathcal{Y}$
- Usually, the returned feedback is **NOT** the optimal (unobservable) label

$$\mathbf{y_t^*} := \text{argmax}_{y \in \mathcal{Y}} U(\mathbf{x_t}, \mathbf{y})$$

- Thus, we model settings where:
  - The user searches using various tools (i.e. query reformulation, browsing)
  - The user cannot manually optimize the argmax
- Model assumes that reliable preference feedback can be derived from observable user behavior

Quantifying User Feedback

- Need to quantify how much improvement $\bar{y}_t$ provides in the utility space
- Not needed for algorithm, but necessary for theoretical analysis

- Simplest case: strictly α-informative
  - α ∈ (0, 1] is an unknown parameter

$$U(x_t, \bar{y}_t) - U(x_t, y_t) \geq \alpha(U(x_t, y_t^*) - U(x_t, y_t))$$

- Utility of $\bar{y}_t$ is higher than that of y_t by a fraction α of the max possible utility range

Quantifying User Feedback

- Violations of the above feedback model are allowed by introducing slack variables $\xi_t \geq 0$
  - Quantifies to what extent the strict α-informative modeling assumption is violated

$$U(x_t, \bar{y_t}) - U(x_t, y_t) \geq \alpha(U(x_t, y_t^*) - U(x_t, y_t)) - \xi_t$$

- Refer to this model as simply $\alpha\text{-informative}$ feedback

- Our regret bounds in the next sections will contain the term ξ_t and α
  - Note that we can express feedback of any quality, even the strict case by choosing $\xi_t = 0$

Quantifying User Feedback

- Now we consider the expected α-informative feedback
  - Even weaker feedback model
  - Positive utility gain is only achieved in expectation over user actions

$$\mathbf{E}_t\left[U(x_t, \bar{y}_t) - U(x_t, y_t)\right] \geq \alpha(U(x_t, y_t^*) - U(x_t, y_t)) - \bar{\xi}_t$$

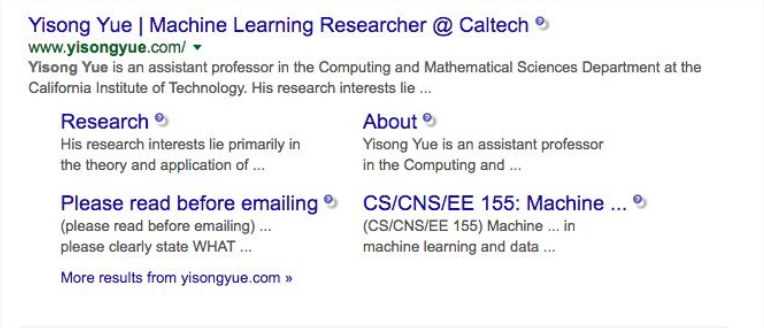
- Expectation is over the user's choice of $\bar{y}_t$ given y_t under context x_t
  - i.e. under a distribution $\mathbf{P}_{x_t}[\bar{\mathbf{y}}_\mathbf{t} \mid \mathbf{y}_\mathbf{t}]$
- Allows for analysis of expected regret

User Study: Preferences from Clicks

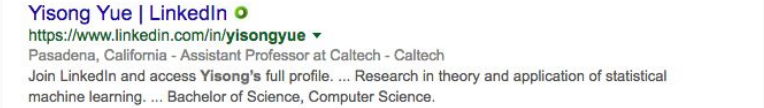
- Experimentally validate that users actually exhibit a preference for predictions with higher utility (implicit feedback is reflective of changes in utility)
- Preference feedback is from clicks in web-search
- Asked subjects to answer 10 questions using Google search
  - Google results: $\mathbf{y}$
  - User feedback (links clicked): $\bar{\mathbf{y}}$
  - Relevance of each document: $r(\mathbf{x}, \mathbf{y}[i])$
    - Manually ranked by assessors, reflects ground-truth utility of link
    - $r(\mathbf{x}, d) \in [0...5]$

Who is Yisong Yue?

A



B



C

D

E

Google $(\mathbf{y})$	A	B	C	D	E
$r(\mathbf{x}, \mathbf{y}[i])$	4	5	3	2	3

User Clicks	B	A	E		
Feedback Vector $(\bar{\mathbf{y}})$	B	A	E	C	D
$r(\mathbf{x}, \bar{\mathbf{y}}[i])$	5	4	3	3	2

Compare Utilities

$$DCG@10(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{10} \frac{r(\mathbf{x}, \mathbf{y}[i])}{\log i + 1}$$

- Need to assess whether $U(x, \bar{y}_t) > U(x, y_t)$
- Use standard metric of information retrieval quality (Manning et al., 2008)

Google $(\mathbf{y})$	A	B	C	D	E
$r(\mathbf{x}, \mathbf{y}[i])$	4	5	3	2	3

DCG = 10.38

Feedback Vector $(\bar{\mathbf{y}})$	B	A	E	C	D
$r(\mathbf{x}, \bar{\mathbf{y}}[i])$	5	4	3	3	2

DCG = 10.82

Three Experimental Conditions

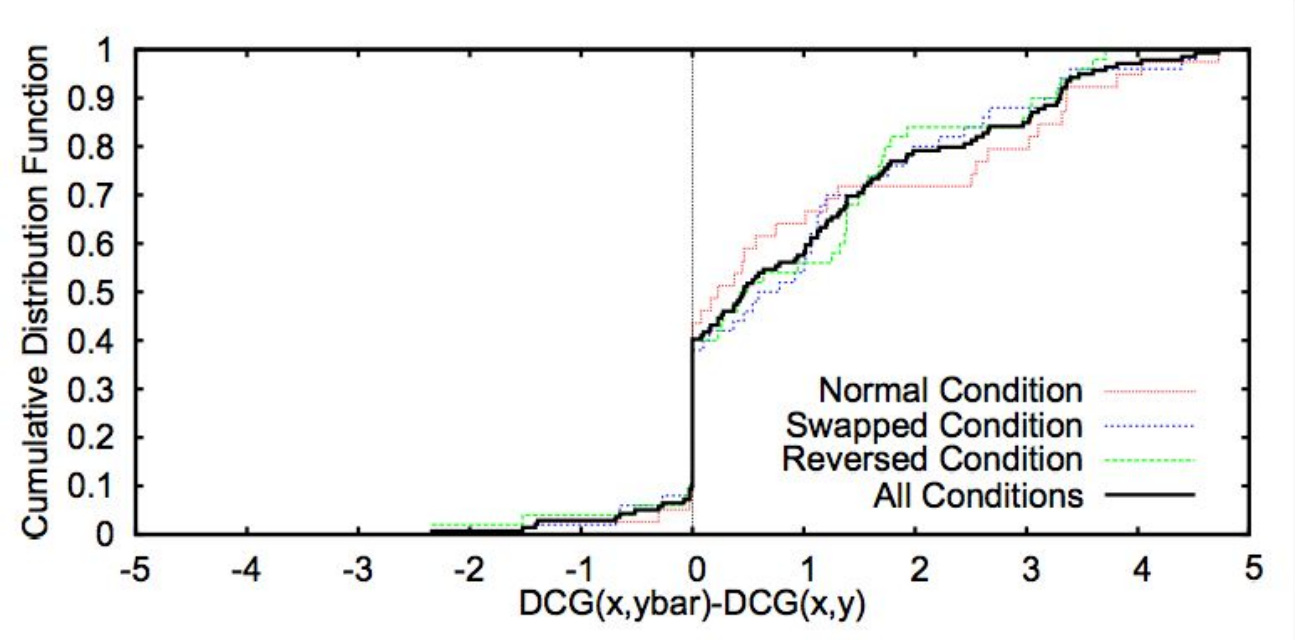
- Also check whether quality of feedback affected by quality of current prediction
  - How $\mathbf{y}$ influences $\bar{\mathbf{y}}$
- Normal
  - Top 10 google results in order
- Reversed
  - Top results in reverse order
- Swapped
  - Top 2 results switched

Google (Normal)	A	B	C	D	E
$r(\mathbf{x}, \mathbf{y}[i])$	4	5	3	2	3

Google (Reversed)	E	D	C	B	A
$r(\mathbf{x}, \mathbf{y}[i])$	3	2	3	5	4

Google (Swapped)	B	A	C	D	E
$r(\mathbf{x}, \mathbf{y}[i])$	5	4	3	2	3

Results



- All CDF's shifted to right
  - Implicit feedback can indeed produce improvements in utility
  - Statistically significant

Results

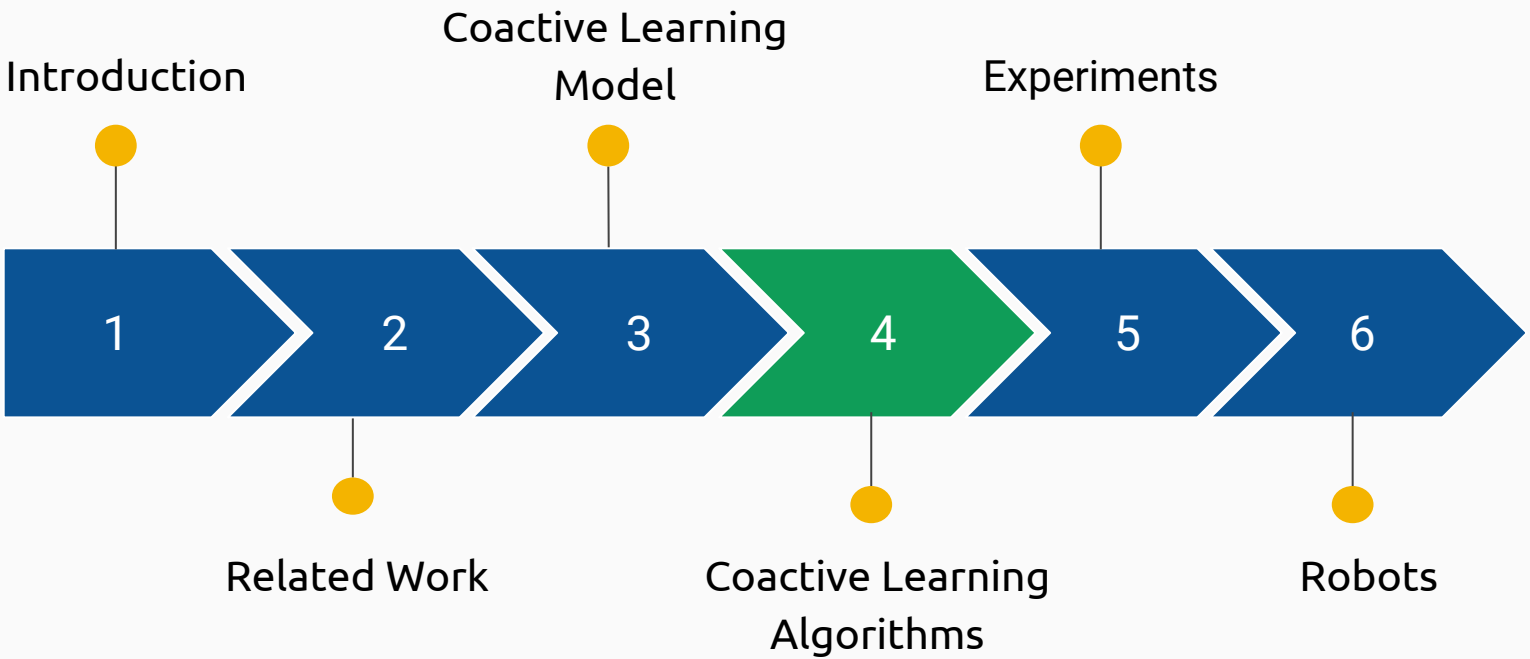
- Additionally, the previous graph shows that users provide accurate preferences across a range of retrieval qualities (normal, random, swapped)

- Intuitively, a worse retrieval system may make it harder to find good results, but it also makes an easier baseline to improve upon

- This intuition is captured by α-informative feedback

- Tradeoff between α and ξ is application-specific
  - The following algorithms do not require knowledge of α or ξ

Roadmap


Introduction
Coactive Learning Model
Experiments
1
2
3
4
5
6
Related Work
Coactive Learning Algorithms
Robots

Modeling Utility

$$U(x, y) = w_*^T \phi(x, y)$$

- Linear function of $\phi(x, y)$ parameterized by w_*^T
- $\phi(x, y)$: feature map dependent on x (context) and y (prediction)
- w_*^T: optimal weight vector

Example: Robot Manipulation

$$U(x, y) = w_*^T \phi(x, y)$$

- Context (x): Environment (positions of other objects, etc)
- Prediction (y): Trajectory (list of waypoints)
- $\phi(x, y)$
  - Features such as distance from each waypoint to objects in environment
  - Can capture interaction between x and y

Preference Perceptron

Algorithm 1 Preference Perceptron.

Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** T **do**
  Observe $\mathbf{x}_t$
  Present $\mathbf{y}_t \leftarrow \text{argmax}_{\mathbf{y} \in \mathcal{Y}} \mathbf{w}_t^\top \phi(\mathbf{x}_t, \mathbf{y})$
  Obtain feedback $\bar{\mathbf{y}}_t$
  Update: $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t)$
end for

- Goal: learn optimal weight vector w_*^T
- Update weight vector using difference in feature representations

Regret Bound of Preference Perceptron

$$REG_T = \frac{1}{T}\sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t) \leq \frac{1}{\alpha T}\sum_{t=1}^{T} \xi_t + \frac{2R\|w^*\|}{\alpha\sqrt{T}}$$

- α is parameter governing assumption that user feedback is α- informative
- Recall: User feedback is α-informative, with slack variables ξ_t:

$$U(x_t, \bar{y_t}) - U(x_t, y_t) \geq \alpha(U(x_t, y_t^*) - U(x_t, y_t)) - \xi_t$$

- R is upper bound on $\|\phi\|$ $(\|\phi(x, y)\| \leq R)$

Interpreting the Regret Bound

$$REG_T = \frac{1}{T}\sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t) \leq \frac{1}{\alpha T}\sum_{t=1}^{T} \xi_t + \frac{2R\|w^*\|}{\alpha\sqrt{T}}$$

Larger $\alpha \rightarrow$ more informative feedback $\rightarrow$ lower regret

Interpreting the Regret Bound

$$REG_T = \frac{1}{T}\sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t) \leq \frac{1}{\alpha T}\sum_{t=1}^{T} \xi_t + \frac{2R\|w^*\|}{\alpha\sqrt{T}}$$

Larger α → more informative feedback → lower regret

Larger ξ_t → worse violations of α-informative assumption → higher regret

Interpreting the Regret Bound

$$REG_T = \frac{1}{T}\sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t) \leq \frac{1}{\alpha T}\sum_{t=1}^{T} \xi_t + \frac{2R\|w^*\|}{\alpha\sqrt{T}}$$

Larger α → more informative feedback → lower regret

Larger ξ_t → worse violations of α-informative assumption → higher regret

Upper bound on $U(x, y)$

Interpreting the Regret Bound

$$REG_T = \frac{1}{T}\sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t) \leq \frac{1}{\alpha T}\sum_{t=1}^{T} \xi_t + \frac{2R\|w^*\|}{\alpha\sqrt{T}}$$

- If strict α-informative assumption holds, slack terms vanish → $\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$ regret
- Note: algorithm does not know α; it just factors into the analysis

Outline of Proof

$$REG_T = \frac{1}{T}\sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t) \leq \frac{1}{\alpha T}\sum_{t=1}^{T} \xi_t + \frac{2R\|w^*\|}{\alpha\sqrt{T}}$$

1. Show $\|w_{T+1}\|^2 \leq 4R^2T$
2. Upper bound $\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t)$ in terms of $\|w_{T+1}\|$
3. Upper bound $\sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t)$ in terms of $\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t)$

   (using alpha-informative assumption)

1) Bounding $\|\mathbf{w_{T+1}}\|^2$

- Expanding $\|w_{T+1}\|^2$ using our perceptron update rule $w_{T+1} = w_T + (\phi(x_T, \bar{y_T}) - \phi(x_T, y_T))$ gives us:

$$\mathbf{w}_{T+1}^\top \mathbf{w}_{T+1} = \mathbf{w}_T^\top \mathbf{w}_T + 2\mathbf{w}_T^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T)) + (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T)$$

1) Bounding $\|\mathbf{w}_{T+1}\|^2$

$$\mathbf{w}_{T+1}^\top \mathbf{w}_{T+1} = \mathbf{w}_T^\top \mathbf{w}_T + 2\mathbf{w}_T^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T)) + (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T)$$

$$\leq \mathbf{w}_T^\top \mathbf{w}_T + 4R^2 \leq 4R^2 T.$$

From previous slide

Want to show this inequality holds

1) Bounding $\|\mathbf{w}_{T+1}\|^2$

$$\mathbf{w}_{T+1}^\top \mathbf{w}_{T+1} = \mathbf{w}_T^\top \mathbf{w}_T + 2\mathbf{w}_T^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))$$
$$+ (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T)$$
$$\leq \mathbf{w}_T^\top \mathbf{w}_T + 4R^2 \leq 4R^2 T.$$

Negative because our algorithm picked y_t over $\bar{y}_t$

1) Bounding $\|\mathbf{w}_{T+1}\|^2$

$$\mathbf{w}_{T+1}^\top \mathbf{w}_{T+1} = \mathbf{w}_T^\top \mathbf{w}_T + 2\mathbf{w}_T^\top(\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))$$
$$+ (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T)$$
$$\leq \mathbf{w}_T^\top \mathbf{w}_T + 4R^2 \leq 4R^2 T.$$

Negative because our algorithm picked y_t over $\bar{y_t}$

We have $\|\phi(x, y)\| \leq R$
so $\|\phi(x_T, \bar{y_T}) - \phi(x_T, y_T))\| \leq 2R$
(triangle inequality)

1) Bounding $\|\mathbf{w}_{T+1}\|^2$

$$\mathbf{w}_{T+1}^\top \mathbf{w}_{T+1} = \mathbf{w}_T^\top \mathbf{w}_T + 2\mathbf{w}_T^\top(\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))$$
$$+ (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^\top (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T)$$
$$\leq \mathbf{w}_T^\top \mathbf{w}_T + 4R^2 \leq 4R^2 T.$$

Negative because our algorithm picked y_t over $\bar{y_t}$

We have $\|\phi(x, y)\| \leq R$
so $\|\phi(x_T, \bar{y_T}) - \phi(x_T, y_T))\| \leq 2R$
(triangle inequality)

From $w_1 = 0$ and repeated application of the inequality
$w_{T+1}^T w_{T+1} \leq w_T^T w_T + 4R^2$

2) Bounding $\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t)$

Using our previous result, we can find a bound on $\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t)$

$$\mathbf{w}_{T+1}^{\top}\mathbf{w}_* = \mathbf{w}_T^{\top}\mathbf{w}_* + (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^{\top}\mathbf{w}_*$$

$$= \sum_{t=1}^{T} (U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)) . \qquad (9)$$

Update rule multiplied by w_*

2) Bounding $\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t)$

Using our previous result, we can find a bound on $\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t)$

$$\mathbf{w}_{T+1}^{\top}\mathbf{w}_* = \mathbf{w}_T^{\top}\mathbf{w}_* + (\phi(\mathbf{x}_T, \bar{\mathbf{y}}_T) - \phi(\mathbf{x}_T, \mathbf{y}_T))^{\top}\mathbf{w}_*$$

$$= \sum_{t=1}^{T} (U(\mathbf{x}_t, \bar{\mathbf{y}}_t) - U(\mathbf{x}_t, \mathbf{y}_t)). \qquad (9)$$

Update rule multiplied by w_*

$w_*\phi(x_T, y_T)$ is our true utility function, summation follows from iterating equation on first line

2) Bounding $\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t)$

$$\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t) = w_{T+1}^T w_* \leq \|w_{T+1}\| \|w_*\|$$

From prev. slide

$$\|w_{T+1}\|^2 \leq 4R^2T$$

$$\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t) \leq 2R\sqrt{T}\|w_*\|$$

2) Bounding $\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t)$

$$\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t) = w_{T+1}^T w_* \leq \|w_{T+1}\| \|w_*\|$$

Cauchy-Schwarz inequality

$$\|w_{T+1}\|^2 \leq 4R^2T$$

From step 1

$$\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t) \leq 2R\sqrt{T}\|w_*\|$$

3) Bounding $\mathbf{REG_T}$

$$\alpha(U(x_t, y_t^*) - U(x_t, y_t)) - \xi_t \leq U(x_t, \bar{y}_t) - U(x_t, y_t)$$

← α-informative feedback assumption

$$\sum_{t=1}^{T} U(x_t, \bar{y}_t) - U(x_t, y_t) \leq 2R\sqrt{T}\|w_*\|$$

← From step 2

$$\alpha \sum_{t=1}^{T} (U(x_t, y_t^*) - U(x_t, y_t)) - \sum_{t=1}^{T} \xi_t \leq 2R\sqrt{T}\|w_*\|$$

$$REG_T = \frac{1}{T}\sum_{t=1}^{T} U(x_t, y_t^*) - U(x_t, y_t) \leq \frac{1}{\alpha T}\sum_{t=1}^{T} \xi_t + \frac{2R\|w^*\|}{\alpha\sqrt{T}}$$

Comparison to Standard Perceptron

- Standard perceptron (for multi-class classification)
  - Requires true label y_t^*
  - Analyzed in terms of number of mistakes made

- Preference perceptron
  - Uses implicit feedback $\bar{y}_t$
  - Analyzed in terms of utility

Batch Update

- Some applications have high volumes of feedback
  - Might not be possible to do an update after every round
- Consider a variant of Algorithm 1 that makes an update every k iterations
  - Uses $\mathbf{w}_t$ obtained from the previous update until the next update
- It is easy to show the following regret bound for batch updates:

$$REG_T \leq \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t + \frac{2R\|\mathbf{w}_*\|\sqrt{k}}{\alpha\sqrt{T}}.$$

Expected α-informative Feedback

Corollary 3 *Under expected α-informative feedback model, the expected regret (over user behavior distribution) of the preference perceptron algorithm can be upper bounded as follows:*

$$\mathbf{E}[REG_T] \leq \frac{1}{\alpha T}\sum_{t=1}^{T}\bar{\xi}_t + \frac{2R\|\mathbf{w}_*\|}{\alpha\sqrt{T}}. \qquad (10)$$

- Follow the argument of Theorem 1, but take expectations over user feedback

Convex Loss Minimization

- We can generalize our results to minimize convex losses defined on the linear utility differences

- At every time step t, there is an (unknown) convex loss function $c_t : \mathcal{R} \to \mathcal{R}$
  - Determines the loss $c_t(U(x_t, y_t) - U(x_t, y_t^*))$
  - The functions c_t are assumed to be non-increasing
  - Sub-derivatives of the c_t's are assumed to be bounded
    - $c_t'(\theta) \in [-G, 0]$ for all t and for all $\theta \in \mathcal{R}$

- The vector $\mathbf{w}_*$ which determines the utility of y_t under context x_t
  - Assumed from a closed and bounded convex set B whose diameter is |B|

Convex Loss Minimization

Algorithm 2 Convex Preference Perceptron.

Initialize $\mathbf{w}_1 \leftarrow \mathbf{0}$
for $t = 1$ **to** T **do**
  Set $\eta_t \leftarrow \frac{1}{\sqrt{t}}$
  Observe $\mathbf{x}_t$
  Present $\mathbf{y}_t \leftarrow \text{argmax}_{\mathbf{y} \in \mathcal{Y}} \mathbf{w}_t^\top \phi(\mathbf{x}_t, \mathbf{y})$
  Obtain feedback $\bar{\mathbf{y}}_t$
  Update: $\bar{\mathbf{w}}_{t+1} \leftarrow \mathbf{w}_t + \eta_t G(\phi(\mathbf{x}_t, \bar{\mathbf{y}}_t) - \phi(\mathbf{x}_t, \mathbf{y}_t))$
  Project: $\mathbf{w}_{t+1} \leftarrow \arg\min_{\mathbf{u} \in \mathcal{B}} \|\mathbf{u} - \bar{\mathbf{w}}_{t+1}\|^2$
end for

- Main differences from Algorithm 1
  - There is a rate η_t associated with the update at time t
  - After every update, the resulting vector $\bar{w}_{t+1}$ is projected back to the set B

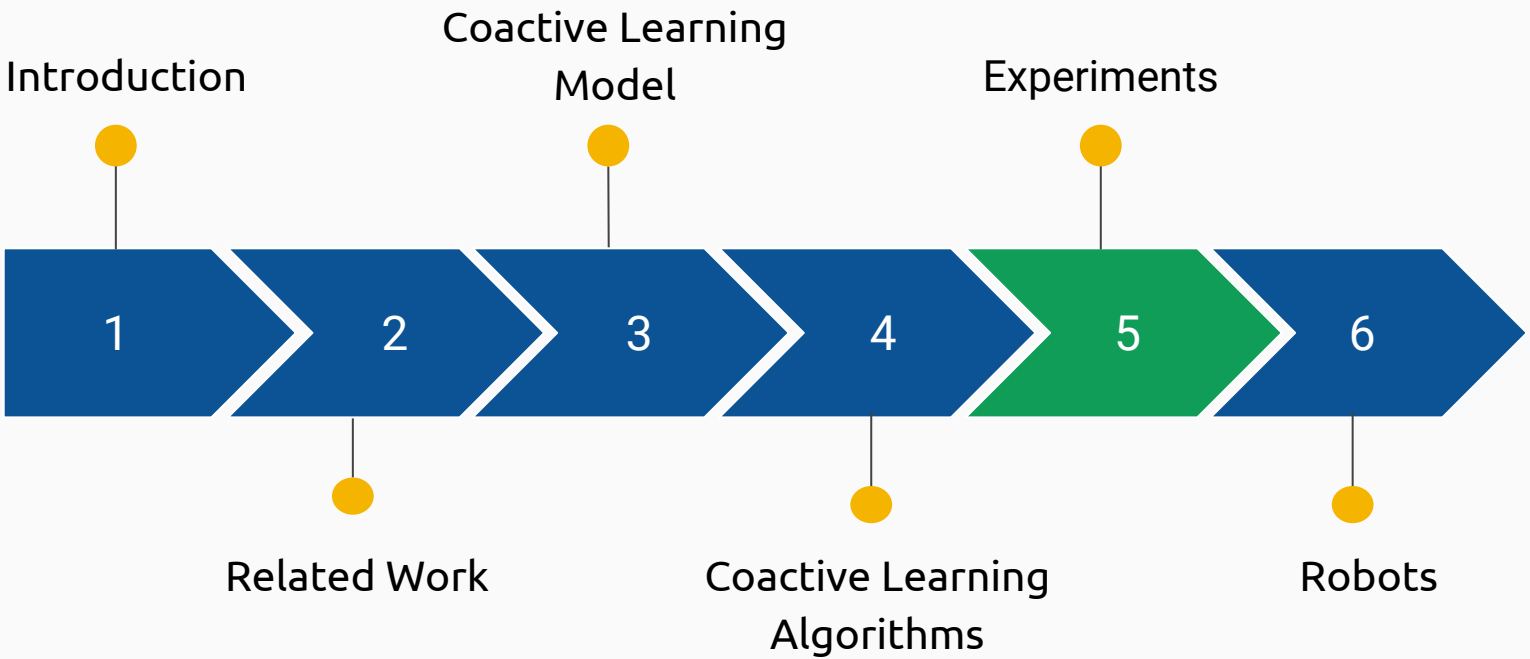
Theorem 4

Theorem 4 *For the convex preference perceptron, we have, for any* $\alpha \in (0, 1]$ *and any* $\mathbf{w}_* \in \mathcal{B}$,

$$\frac{1}{T}\sum_{t=1}^{T} c_t(U(\mathbf{x}_t, \mathbf{y}_t) - U(\mathbf{x}_t, \mathbf{y}_t^*)) \leq \frac{1}{T}\sum_{t=1}^{T} c_t(0) + \frac{2G}{\alpha T}\sum_{t=1}^{T}\xi_t + \frac{1}{\alpha}\left(\frac{|\mathcal{B}|G}{2\sqrt{T}} + \frac{|\mathcal{B}|G}{T} + \frac{4R^2G}{\sqrt{T}}\right). \quad (11)$$

- Main differences from Theorem 1
  - $c_t(0)$ is the minimum possible convex loss
  - Under strict α-informative feedback, average loss approaches best achievable loss: $\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$
    - Larger constant factors than Theorem 1

Roadmap


Introduction
Coactive Learning Model
Experiments
1
2
3
4
5
6
Related Work
Coactive Learning Algorithms
Robots

Testing

- Empirically evaluate Preference Perceptron
  - Structured objects (rankings)

- Strong vs Weak Feedback
  - See how regret of algorithm changes with feedback quality
  - Different levels of α-informativity

- Noisy feedback
  - Directly uses user feedback
  - Compare to SVM

Learning to Rank

- Evaluated perceptron on Yahoo! Learning to rank dataset
- Query-url feature vectors
  - $\mathbf{x}_i^q$ for query *q* and URL *i*
  - Relevance rating $r_i^q \in [0, 4]$
  - $\mathbf{y}_i$: index of URL at position *i* in the ranking
- Joint feature map: $$\mathbf{w}^\top \phi(q, \mathbf{y}) = \sum_{i=1}^{5} \frac{\mathbf{w}^\top \mathbf{x}_{\mathbf{y}_i}^q}{\log(i+1)}$$

Procedure

- Query q_t at time t
- Perceptron presents ranking $\mathbf{y}_t^q$ that maximizes $\mathbf{w}_t^T \phi(q_t, \mathbf{y})$
  - Equivalent to sorting URL's by $\mathbf{w}_t^T \mathbf{x}_i^{\mathbf{q}_t}$
- Utility regret:

$$\frac{1}{T} \sum_{t=1}^{T} \mathbf{w}_*^\top \left(\phi(q_t, \mathbf{y}^{q_t *}) - \phi(q_t, \mathbf{y}^{q_t}) \right)$$

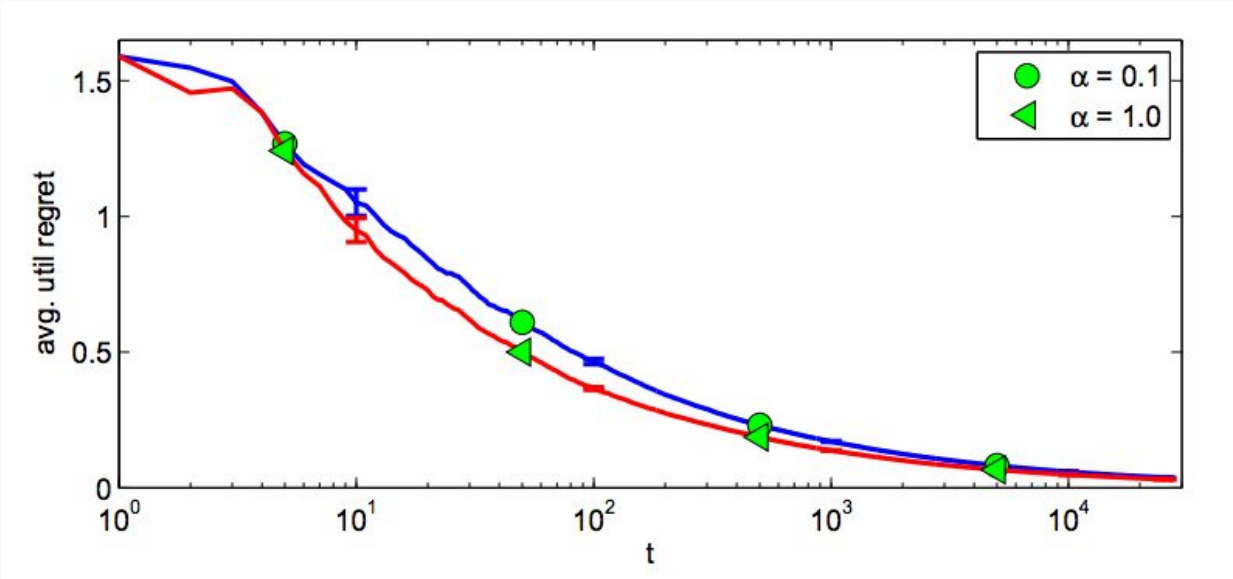
Strong vs. Weak Feedback

- Goal: see how regret of algorithm changes with feedback quality
  - Different values of α

- Given predicted ranking $\mathbf{y}_t$, user goes down list until they find URL's such that resulting $\bar{\mathbf{y}}_t$ satisfies α-informativity

- Update $\mathbf{w}_{t+1}$ and repeat

Strong vs. Weak Feedback

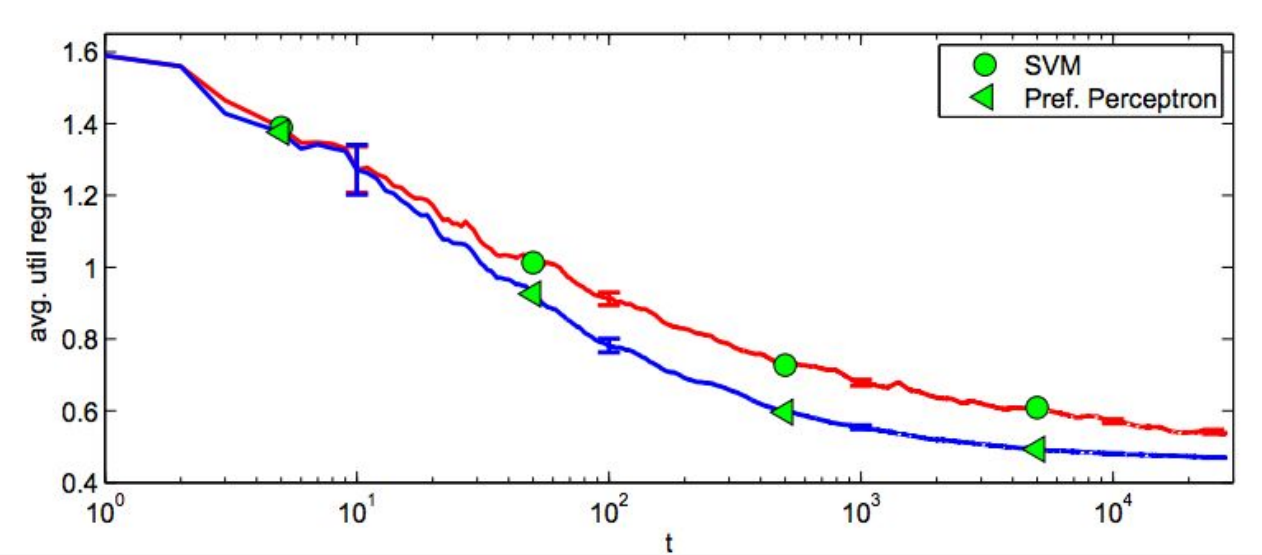


- Regret with α = 1.0 less than with α = 0.1
- Difference less than factor of 10

Noisy Feedback

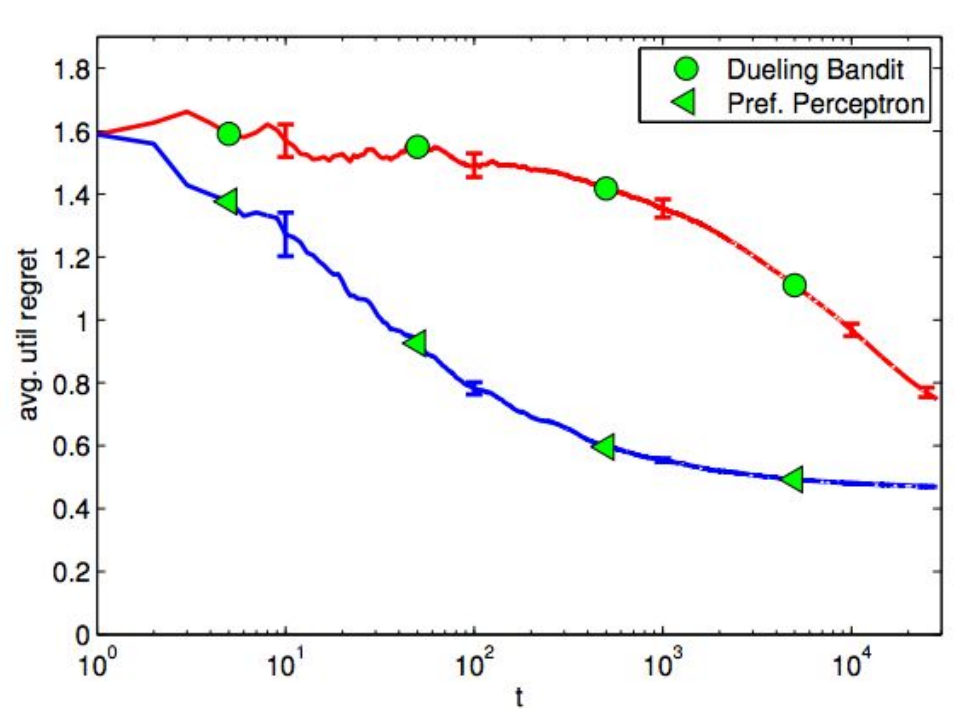
- Previous experiment: feedback based on actual utility values from optimal $\mathbf{w}_*$
- Goal: use actual relevance labels from users
- Produces noisy feedback
  - No linear model can perfectly fit relevance labels

Noisy Feedback



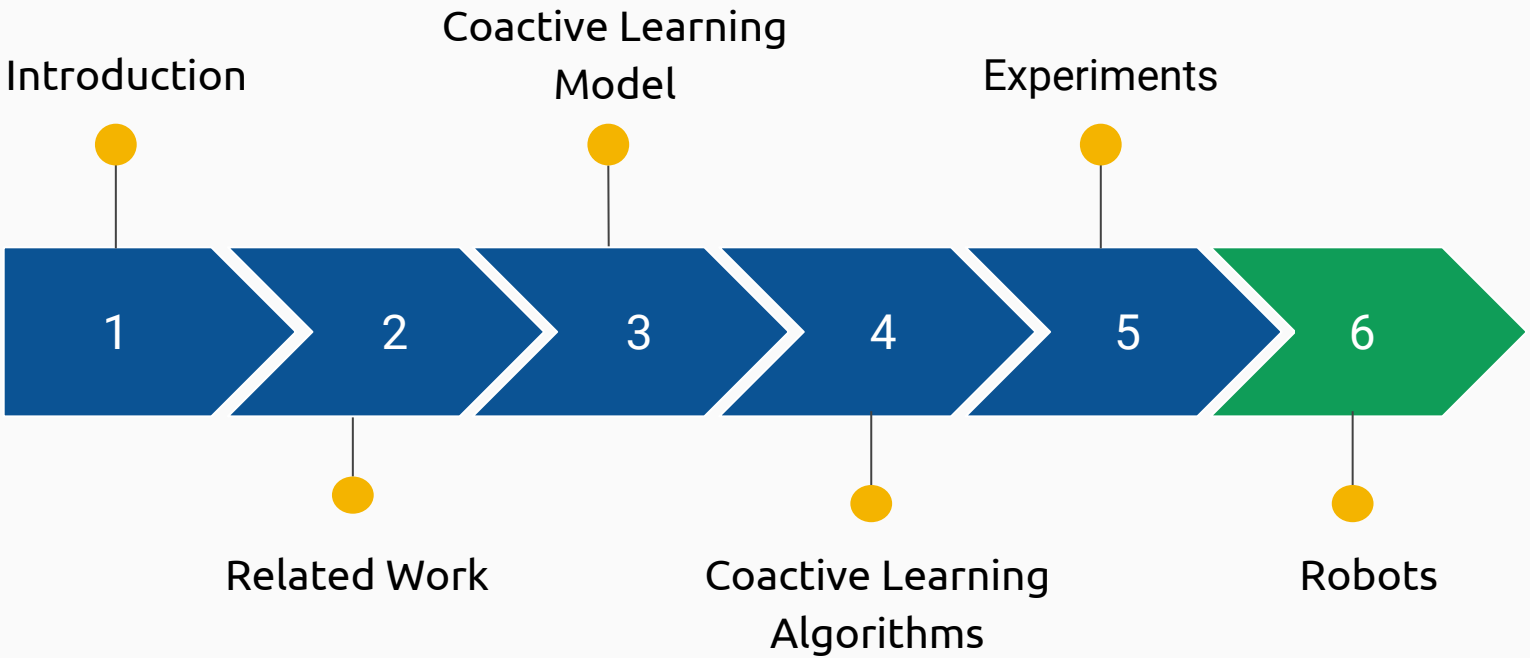
- Regret with Preference Perceptron significantly less than with SVM
- Preference Perceptron runs orders of magnitude more quickly
- Regret converges to non-zero value

Comparison with Dueling Bandits



- Performs better than dueling bandits

Roadmap


Introduction
Coactive Learning Model
Experiments
1
2
3
4
5
6
Related Work
Coactive Learning Algorithms
Robots

ROBOTS!

- Teaching a robot to produce desired motions has been a long standing goal

- Past research has focused on mimicking expert's demonstrations
  - Autonomous helicopter flights
  - Ball-In-A-Cup Experiment

- Applicable to scenarios when it is clear to an expert what constitutes a good trajectory
  - Extremely challenging in some scenarios, especially involving high DoF manipulators
  - Users have to give
    - End-effector's location at each time-step
    - Full configuration of the arm in a way that is spatially and temporally consistent

Video - Robotic Application

http://www.youtube.com/watch?v=uLktpkd7ojA

Robots + Coactive Learning

- User never discloses optimal trajectory (or provides optimal feedback) to the robot
- Robot learns preferences from sub-optimal suggestions on how trajectory can be improved
- Authors design appropriate features that consider
  - Robot Configurations
  - Object-object relations
  - Temporal behavior
- Learns score functions reflecting user preferences from implicit feedback

Robot Learning Model

Goal: Learn user preferences

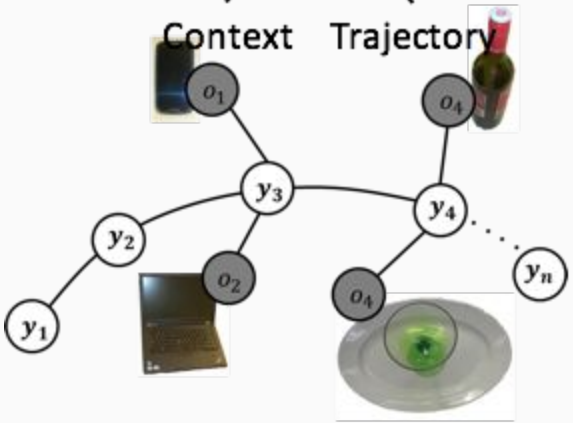
Scoring Function

$$s(x, y; w) = \boxed{w_O^T \phi_O(x, y)} + \boxed{w_E^T \phi_E(x, y)}$$

Context Trajectory

Object-object Interactions

Robot configuration and Environment Interactions



Trajectory graph

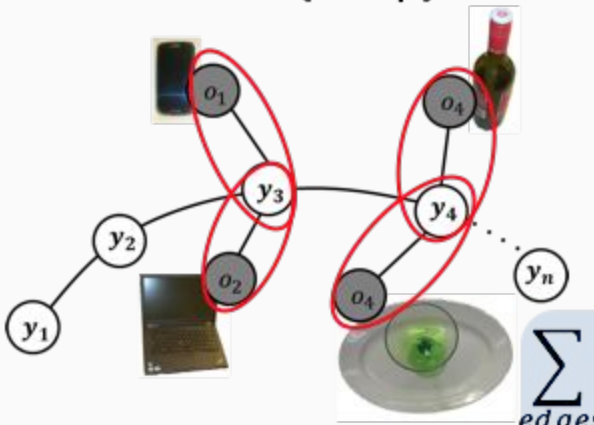
Connecting waypoints to neighboring objects

Features

Object attributes: {electronic, fragile, sharp, liquid, hot, …}

E.g. Laptop: {electronic, fragile}
Knife: {sharp} …..

Hermans et. al. ICRA w/s 2011
Koppula et. al. NIPS 2011



Trajectory graph

Object-object Interactions

$$w_O^T \phi_O(x, y)$$

$$\sum_{edges} \sum_{l,k \in labels} \mathbf{1}(edge, l, k) w_{lk}^T \phi_{o-o}(x, y; edge)$$

Distance features

Features

$$s(x, y; w) = w_O^T \phi_O(x, y) + w_E^T \phi_E(x, y)$$

Features $\in \mathbb{R}^{75}$

Object-object Interactions

Robot configuration and Environment Interactions

1. Spectrogram
2. Object's distance from horizontal and vertical surfaces
3. Object's angle with vertical axis
4. Robot's wrist and elbow configuration in cylindrical co-ordinate

Computing Trajectory Rankings

- For a given task with context x, we would like to maximize the current scoring function

$$y^* = \arg\max_y s(x, y; w_O, w_E).$$

- Trajectory space is continuous and needs to be discretized to maintain argmax tractable
  - We can sample trajectories from the continuous space
    - Rapidly Exploring Random Tree (RRT)

- We can sort the trajectories by their trajectory scores to find the argmax

Trajectory Preference Perceptron (TPP)

Algorithm 1 Trajectory Preference Perceptron. (TPP)

Initialize $w_O^{(1)} \leftarrow 0, w_E^{(1)} \leftarrow 0$
for $t = 1$ to T **do**
  Sample trajectories $\{y^{(1)}, ..., y^{(n)}\}$
  $y_t = argmax_y s(x_t, y; w_O^{(t)}, w_E^{(t)})$
  Obtain user feedback $\bar{y}_t$
  $w_O^{(t+1)} \leftarrow w_O^{(t)} + \phi_O(x_t, \bar{y}_t) - \phi_O(x_t, y_t)$
  $w_E^{(t+1)} \leftarrow w_E^{(t)} + \phi_E(x_t, \bar{y}_t) - \phi_E(x_t, y_t)$
end for

- Almost the same as algorithm in previous paper except we sample trajectories
- Proof can be adapted to show that the expected average regret is upper-bounded by

$$E[REG_T] \leq \mathcal{O}(\frac{1}{\alpha\sqrt{T}} + \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t)$$

Experiment

- Evaluate approach on 16 pick-and-place robotic tasks in a grocery store checkout setting
- For each task, train and test on scenarios with different objects being manipulated and/or with a different environment
- An expert labeled 1300 trajectories on a Likert scale of 1-5 (where 5 is the best) on the basis of subjective human preferences
- Evaluate quality of trajectories after robot has grasped the items and while it moves them for checkout

Baseline Algorithms

- *Geometric*: It plans a path, independent of the task, using a BiRRT planner
- *Manual*: It plans a path following certain manually coded preferences
- *Oracle-svm*: This algorithm leverages the expert's labels on trajectories and is trained using SVM-rank in a batch manner
- *MMP-online*: This is an online implementation of Maximum margin planning (MMP)
- *TPP*: This is the algorithm from the paper
- Where applicable, algorithms are applied to two different settings
  a. *Untrained* setting: algorithm learns preferences for the new task from scratch without observing any previous feedback
  b. *Pretrained* setting: algorithms are pre-trained on other similar tasks, and then adapt to the new task

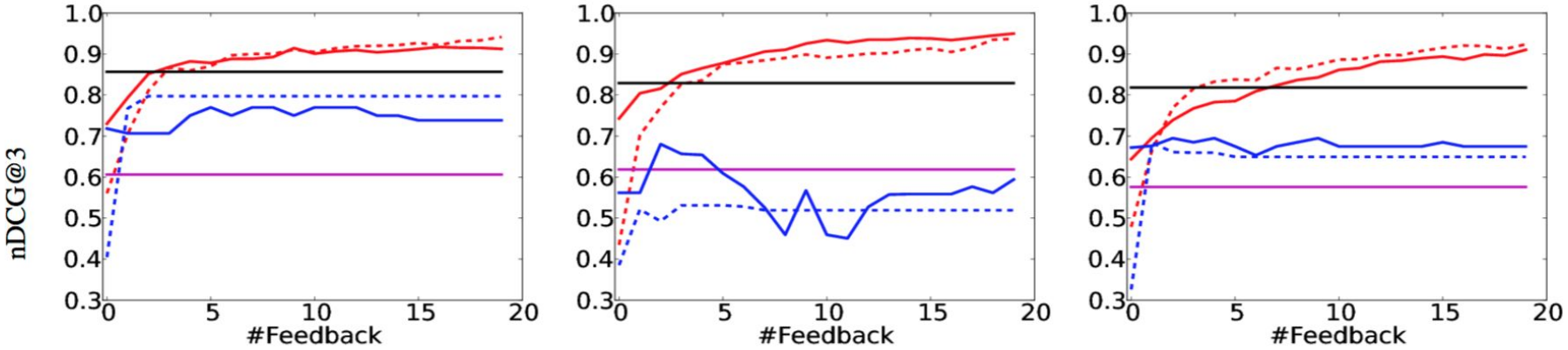
Results

Table 1: **Comparison of different algorithms and study of features in untrained setting.** Table contains average nDCG@1(nDCG@3) values over 20 rounds of feedback.

	Algorithms	Manipulation centric	Environment centric	Human centric	Mean
	Geometric	0.46 (0.48)	0.45 (0.39)	0.31 (0.30)	0.40 (0.39)
	Manual	0.61 (0.62)	0.77 (0.77)	0.33 (0.31)	0.57 (0.57)
TPP Features	Obj-obj interaction	0.68 (0.68)	0.80 (0.79)	0.79 (0.73)	0.76 (0.74)
TPP Features	Robot arm config	0.82 (0.77)	0.78 (0.72)	0.80 (0.69)	0.80 (0.73)
TPP Features	Object trajectory	0.85 (0.81)	0.88 (0.84)	0.85 (0.72)	0.86 (0.79)
TPP Features	Object environment	0.70 (0.69)	0.75 (0.74)	0.81 (0.65)	0.75 (0.69)
	TPP (all features)	**0.88 (0.84)**	**0.90 (0.85)**	**0.90 (0.80)**	**0.89 (0.83)**
	MMP-online	0.47 (0.50)	0.54 (0.56)	0.33 (0.30)	0.45 (0.46)

- TPP performs better than baseline algorithms
- All features combined together give the best performance

Results



(a) Same environment, different object (b) New Environment, same object (c) New Environment, different object

Figure 5: Study of generalization with change in object, environment and both. Manual, Oracle-SVM, Pre-trained MMP-online (—), Untrained MMP-online (– –), Pre-trained TPP (—), Untrained TPP (– –).

Experiment: User Study

- Five users used system to train Baxter for grocery checkout tasks
- A set of 10 tasks of varying difficulty level was presented to users one at a time
- Users were instructed to provide feedback until they were satisfied with the top ranked trajectory
  - Zero-G was provided kinesthetically on the robot
  - Re-rank was elicited in a simulator
- To quantify the quality of learning each user evaluated on a Likert scale of 1-5 (5 is the best)
  - Their own trajectories (self score)
  - The trajectories learned of the other users (cross score)
  - Those predicted by Oracle-svm
- Time a user took for each task was also recorded

User Feedback

Re-rank

Zero-G

Results: User Study

- Within 5 feedbacks the users were able to improve over Oracle-svm
- Re-rank feedback was popular for easier tasks
- As difficulty increased the users relied more on zero-G feedback
- Each user took on average 5.5 minutes per-task
- Shows algorithm is realizable in practice on high DoF manipulators

Table 2: Shows learning statistics for each user averaged over all tasks. The number in parentheses is standard deviation.

User	# Re-ranking feedback	# Zero-G feedback	Average time (min.)	Trajectory Quality self	Trajectory Quality cross
1	5.4 (4.1)	3.3 (3.4)	7.8 (4.9)	3.8 (0.6)	4.0 (1.4)
2	1.8 (1.0)	1.7 (1.3)	4.6 (1.7)	4.3 (1.2)	3.6 (1.2)
3	2.9 (0.8)	2.0 (2.0)	5.0 (2.9)	4.4 (0.7)	3.2 (1.2)
4	3.2 (2.0)	1.5 (0.9)	5.3 (1.9)	3.0 (1.2)	3.7 (1.0)
5	3.6 (1.0)	1.9 (2.1)	5.0 (2.3)	3.5 (1.3)	3.3 (0.6)

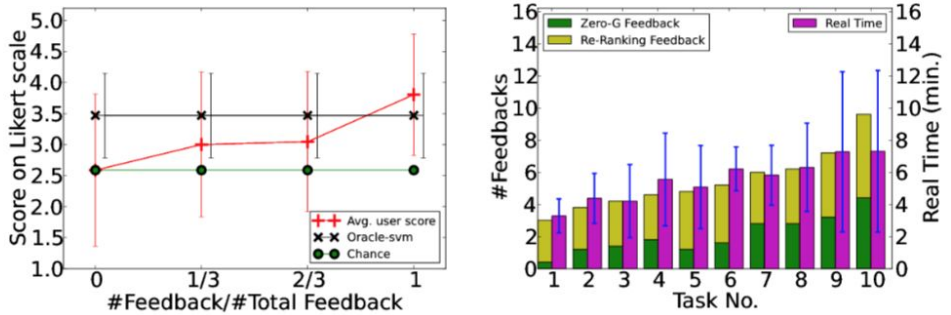


Figure 6: **(Left)** Average quality of the learned trajectory after every one-third of total feedback. **(Right)** Bar chart showing the average number of feedback and time required for each task. Task difficulty increases from 1 to 10.

Any Questions?

Citations

- Shivaswamy, P., Joachims, T. (2015). Coactive Learning
- Jain, A., et al. (2013). Learning Trajectory Preferences for Manipulators via Iterative Improvement
- Cornell University Robotics Learning Lab (http://pr.cs.cornell.edu/coactive/)

Extra Slides

Robot Learning Model

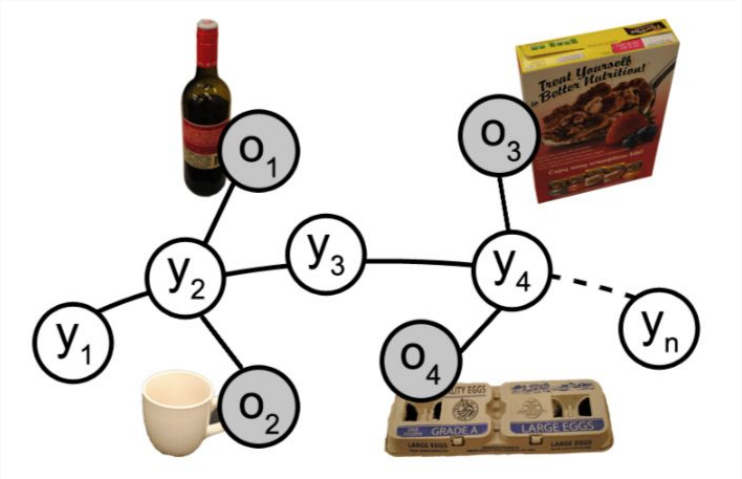
- The robot is given a context x
  - Describes the environment, the objects, and any other input relevant to the problem
- The robot has to figure out what is a good trajectory y for this context
- We assume that the user has a scoring function $s^*(x,y)$
  - Reflects how much he values each trajectory y for context x
  - Higher score → better trajectory
  - Scoring function cannot be observed directly
  - User can provide us with *preferences* that reflect this scoring function
- The robot's goal is to learn a function $s(x, y; w)$
  - Approximates the user's true scoring function $s^*(x, y)$ as closely as possible

Trajectory Features

- We compute features capturing robot's arm configuration
  - Location of its elbow and wrist, w.r.t. to its shoulder, in cylindrical coordinate system, (r,θ,z)
- Features to capture orientation and temporal behavior of the object to be manipulated
  - Cosine of the object's maximum deviation, along the vertical axis, from its final orientation at the goal location
  - Spectrogram for each one-third part for the movement of the object in x, y, z directions
  - Compute the average power spectral density in the low and high frequency part
- Features for object-environment
  - Captures temporal variation of vertical and horizontal distances from its surrounding surfaces
  - Minimum vertical distance from the nearest surface below it
  - Minimum horizontal distance from the surrounding surfaces
  - Minimum distance from the table, on which the task is being performed
  - Minimum distance from the goal location
  - Feature from time-frequency spectrogram of object's vertical distance from the nearest surface below it

Features for Object-Object Interactions

- We enumerate waypoints of trajectory y as $y_1,..,y_N$
- Objects in the environment as $O = \{o_1, .., o_K\}$
- The robot manipulates the object $\bar{o} \in O$
- A few of the trajectory waypoints would be affected by the other objects in the environment
  - We connect an object o_k to a trajectory waypoint
    - If the minimum distance to collision is less than a threshold
    - If o_k lies below $\bar{o}$
  - The edge connecting y_j and o_k is denoted as $(y_j,o_k) \in E$



Features for Object-Object Interactions

- Attributes of an object determine trajectory quality
  - For every object o_k, we consider a vector of M binary variables $[l_k^1, .., l_k^M]$
  - Each $l_k^m = \{0, 1\}$ indicating whether object o_k possesses property m or not

- If the set of possible properties are {heavy, fragile, sharp, hot, liquid, electronic}, then
  - Laptop can have labels [0, 1, 0, 0, 0, 1]
  - Glass Table can have labels [0, 1, 0, 0, 0, 0]

- For every (y_j, o_k) edge, we extract following four features $\varphi_{oo}(y_j, o_k)$:
  - Projection of minimum distance to collision along x, y and z (vertical) axis
  - Binary variable, that is 1, if o_k lies vertically below $\bar{o}$, 0 otherwise

- We now define the score $s_O(\cdot)$ over this graph as follows:

$$s_O(x, y; w_O) = \sum_{(y_j, o_k) \in \mathcal{E}} \sum_{p,q=1}^{M} l_k^p l^q [w_{pq} \cdot \phi_{oo}(y_j, o_k)]$$

Robot Scoring Function

- We model the user's scoring function with the following parameterized family of functions

$$s(x, y; w) = w \cdot \phi(x, y)$$

- We further decompose the score function in two parts
  - s_O - Objects the trajectory is interacting with
  - s_E - Object being manipulated and the environment

$$s(x, y; w_O, w_E) = s_O(x, y; w_O) + s_E(x, y; w_E) = w_O \cdot \phi_O(x, y) + w_E \cdot \phi_E(x, y)$$

Atomic Predictions

- Movie recommendations from MovieLens dataset
  - 3090 movies rated by 6040 users
  - Over 1 million pairs

- Split data in half
  - First set:
    - Obtain feature vector $\mathbf{m}_j$ for each movie *j* using SVD
  - Second set:
    - Consider problem of recommending movies based on features $\mathbf{m}_j$

- Simulates recommending movies to new user based on movie features from old users

Procedure

- For each user *i*
  - Best least-squares approximation $\mathbf{w}_{i*}^T \mathbf{m}_j$ to user utility

- At each *t*
  - Best available movie: $\mathbf{m}_{t^*}$
  - Recommended movie: $\mathbf{m}_t$
  - User reveals preference
  - Recommended movie and feedback movie removed from subsequent set of candidate movies

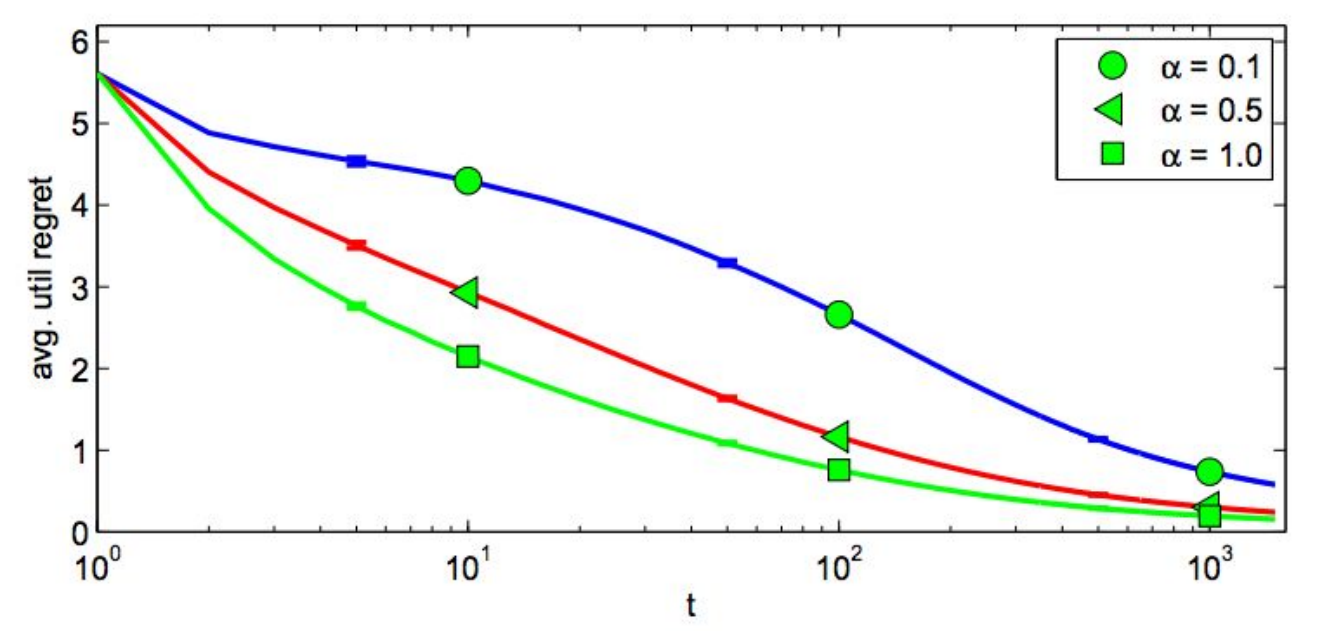
- Utility regret:

$$\frac{1}{T} \sum_{t=1}^{T} \mathbf{w}_{i*}^{\top} (\mathbf{m}_{t*} - \mathbf{m}_t)$$

Strong vs. Weak Feedback

- Goal: see how regret of algorithm changes with feedback quality
  - Different values of α
- We recommend movie with maximum utility from current $\mathbf{w}_t$
- User returns a movie with smallest utility that satisfies α-informativity
- Update $\mathbf{w}_{t+1}$ and repeat
- Regret calculated at each step and averaged over all users

Strong vs. Weak Feedback

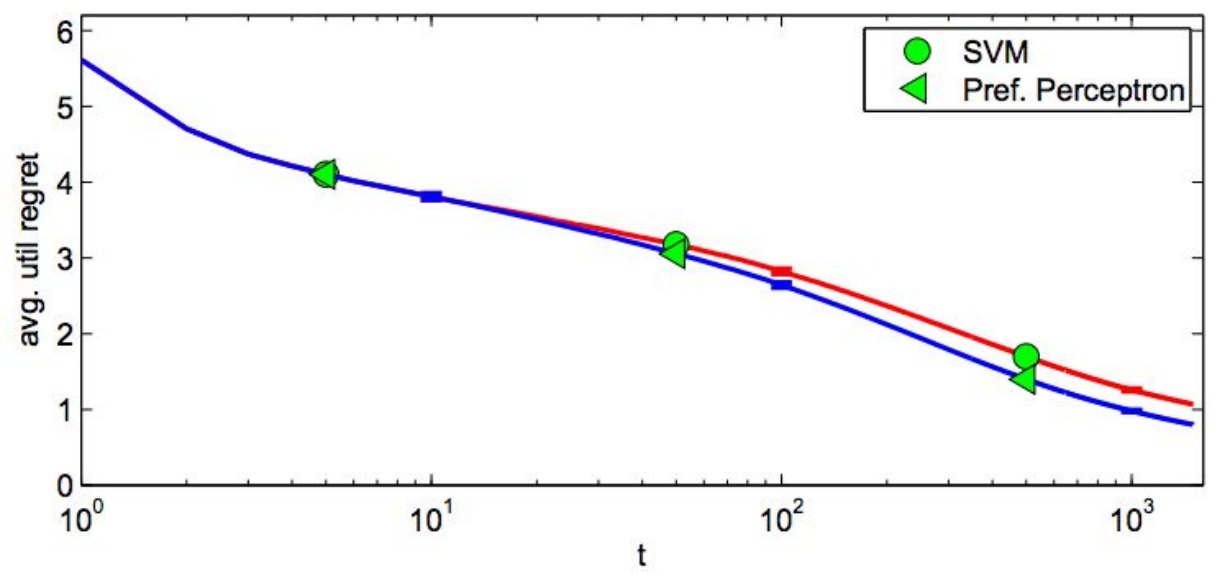


- Regret with α = 1.0 less than with α = 0.5 and α = 0.1
- Difference less than factor of 2 and 10, respectively
- Regret converges faster for higher values of α

Noisy Feedback

- Previous experiment: feedback based on actual utility values from optimal $\mathbf{w}_*$
- Goal: use actual feedback from users
- User receives recommendation
- User returns a movie one rating higher

Noisy Feedback



- Regret with Preference Perceptron significantly less than with SVM
- Preference Perceptron runs orders of magnitude more quickly