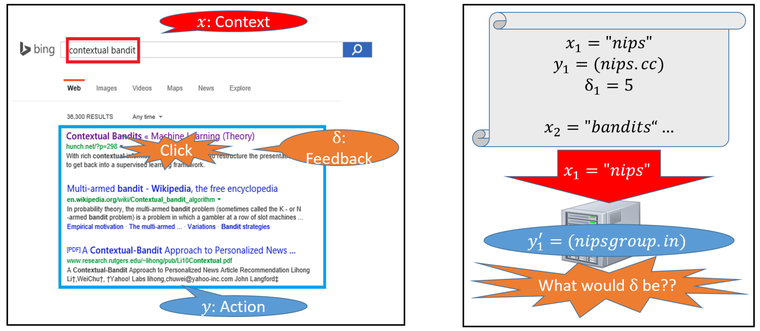
The Self-Normalized Estimator for Counterfactual Learning

Adith Swaminathan and Thorsten Joachims
Department of Computer Science, Cornell University

Cornell University · Neural Information Processing Systems Foundation

Setting: Batch learning from logged bandit feedback

Can we re-use the logs of interactive systems to reliably train them offline?

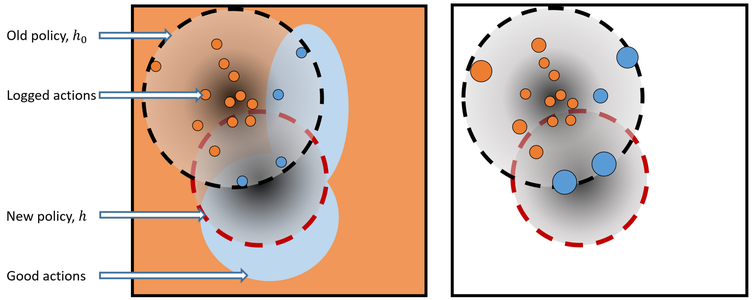


Use $\langle x_i, y_i, \delta_i \rangle_{i=1}^n$ to find a good policy $\boldsymbol{h}(y \mid x)$.

Challenge: Logs are **biased** and **incomplete**.

Approach: Importance sampling

Inject randomization in the system, $y_i \sim h_0(y \mid x_i)$. Log the propensities $p_i = h_0(y_i \mid x_i)$ [5].



$$\underbrace{\hat{R}(h)}_{\text{Risk of new policy } h} = \frac{1}{n}\sum_{i=1}^{n} \underbrace{\delta_i}_{\text{Feedback}} \underbrace{\frac{1}{h_0(y_i \mid x_i)}}_{\text{Propensity, } p_i} \underbrace{h(y_i \mid x_i)}_{\text{New policy}}.$$

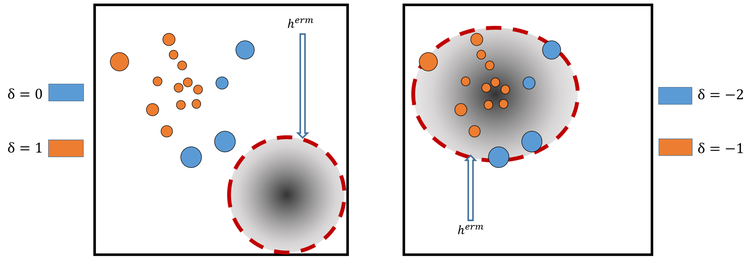
Minimize an upper bound on the empirical risk [2].

$$h^{erm} = \operatorname*{argmin}_{h \in \mathcal{H}} \underbrace{\hat{R}(h)}_{\text{Empirical risk}} + \lambda \underbrace{Reg(h)}_{\text{Regularizer}}.$$

$\hat{R}(h)$ is unbiased but flawed.

Problem		Fix
Unbounded variance.	$\Rightarrow$	Threshold the propensities [4].
Non-uniform variance.	$\Rightarrow$	Use empirical variance regularizers [2].
Propensity overfitting.	$\Rightarrow$	**Deal-breaker.**

Propensity overfitting



Self-Normalized estimator

Idea: Use importance sampling diagnostics to detect overfitting [1].

$$\hat{S}(h) = \frac{1}{n}\sum_{i=1}^{n} \frac{h(y_i \mid x_i)}{p_i}. \qquad \forall h \in \mathcal{H}, \quad \mathbb{E}\left[\hat{S}(h)\right] = 1.$$

Employ $\hat{S}(h)$ as a *multiplicative control variate* to get the self-normalized estimator [6].

$$\hat{R}^{sn}(h) = \left(\sum_{i=1}^{n} \frac{\delta_i h(y_i \mid x_i)}{p_i}\right) \Big/ \left(\sum_{i=1}^{n} \frac{h(y_i \mid x_i)}{p_i}\right).$$

$\hat{R}^{sn}(h)$ is biased but asymptotically consistent.
typically has lower variance than $\hat{R}(h)$.
is **equivariant**.

Norm-POEM: *Normalized* Policy Optimizer for Exponential Models

Exponential Models assume: $h_w(y \mid x) \propto \exp\langle w \cdot \phi(x, y) \rangle$.

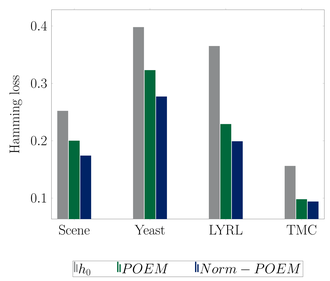
$$w^* = \operatorname*{argmin}_w \quad \hat{R}^{sn}(h_w) \quad + \quad \lambda\sqrt{\frac{\hat{Var}(\hat{R}^{sn}(h_w))}{n}} \quad + \quad \mu\|w\|^2.$$

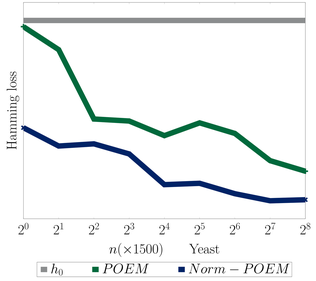
Non-convex optimization over w. Gradient descent (e.g. l-BFGS) still works well.

For a simple Conditional Random Field prototype that can learn from logged bandit feedback to predict structured outputs, please visit
`http://www.cs.cornell.edu/~adith/poem`.

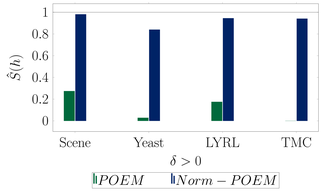
Experiments

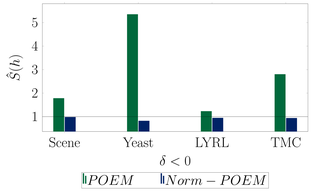
Supervised $\mapsto$ Bandit [3] Multi-Label classification with $\delta \equiv$ Hamming loss on four datasets.





Norm-POEM significantly outperforms the usual estimator (POEM) on all datasets.





Norm-POEM indeed avoids overfitting $\hat{S}(h)$ and is equivariant.

Hamming loss	Norm-IPS	Norm-POEM
Scene	1.072	1.045
Yeast	3.905	3.876
TMC	3.609	2.072
LYRL	0.806	0.799

Time(s)	Scene	Yeast	TMC	LYRL
POEM (l-bfgs)	78.69	98.65	716.51	617.30
Norm-POEM (l-bfgs)	7.28	10.15	227.88	142.50
CRF (scikit-learn)	4.94	3.43	89.24	72.34

Norm-POEM still benefits from variance regularization and is quick to optimize.

Open questions

- What property (apart from **equivariance**) of an estimator ensures good optimization?
- Can we make a more informed bias-variance trade-off when constructing these estimators?
- How can we reliably optimize these objectives at scale?

References

[1] Adith Swaminathan and Thorsten Joachims. The Self-Normalized Estimator for Counterfactual Learning. In *NIPS*, 2015.
[2] Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *ICML*, 2015.
[3] Alina Beygelzimer and John Langford. The offset tree for learning with partial labels. In *KDD*, 2009.
[4] Alexander L. Strehl, John Langford, Lihong Li, and Sham Kakade. Learning from logged implicit exploration data. In *NIPS*, 2010.
[5] Léon Bottou, Jonas Peters, Joaquin Q. Candela, Denis X. Charles, Max Chickering, Elon Portugaly, Dipankar Ray, Patrice Y. Simard, and Ed Snelson. Counterfactual reasoning and learning systems: the example of computational advertising. *Journal of Machine Learning Research*, 14(1):3207–3260, 2013.
[6] Tim Hesterberg. Weighted average importance sampling and defensive mixture distributions. *Technometrics*, 37:185–194, 1995.

Acknowledgment

This research was funded through NSF Award IIS- 1247637, IIS-1217686, IIS-1513692, the JTCII Cornell-Technion Research Fund, and a gift from Bloomberg.