Journal of Machine Learning Research 1 (2015) 1-25 Submitted 6/15; Published

Batch Learning from Logged Bandit Feedback through Counterfactual Risk Minimization

Adith Swaminathan ADITH@CS.CORNELL.EDU
Department of Computer Science
Cornell University
Ithaca, NY 14853, USA

Thorsten Joachims TJ@CS.CORNELL.EDU
Department of Computer Science
Cornell University
Ithaca, NY 14853, USA

Editor: Vladimir N.Vapnik, Alexander Gammerman, Vladimir Vovk

Abstract

We develop a learning principle and an efficient algorithm for batch learning from logged bandit feedback. This learning setting is ubiquitous in online systems (e.g., ad placement, web search, recommendation), where an algorithm makes a prediction (e.g., ad ranking) for a given input (e.g., query) and observes bandit feedback (e.g., user clicks on presented ads). We first address the counterfactual nature of the learning problem (Bottou et al., 2013) through propensity scoring. Next, we prove generalization error bounds that account for the variance of the propensity-weighted empirical risk estimator. In analogy to the Structural Risk Minimization principle of Wapnik and Tscherwonenkis (1979), these constructive bounds give rise to the Counterfactual Risk Minimization (CRM) principle. We show how CRM can be used to derive a new learning method—called Policy Optimizer for Exponential Models (POEM)—for learning stochastic linear rules for structured output prediction. We present a decomposition of the POEM objective that enables efficient stochastic gradient optimization. The effectiveness and efficiency of POEM is evaluated on several simulated multi-label classification problems, as well as on a real-world information retrieval problem. The empirical results show that the CRM objective implemented in POEM provides improved robustness and generalization performance compared to the state-of-the-art.

Keywords: empirical risk minimization, bandit feedback, importance sampling, propensity score matching, structured prediction

1. Introduction

Log data is one of the most ubiquitous forms of data available, as it can be recorded from a variety of systems (e.g., search engines, recommender systems, ad placement) at little cost. The interaction logs of such systems typically contain a record of the input to the system (e.g., features describing the user), the prediction made by the system (e.g., a recommended list of news articles) and the feedback (e.g., number of ranked articles the user read) (Li et al., 2010). The feedback, however, provides only partial information— "bandit feedback"— limited to the particular prediction shown by the system. The feedback for

all the other predictions the system could have made is typically not known. This makes learning from log data fundamentally different from supervised learning, where "correct" predictions (e.g., the best ranking of news articles for that user) together with a loss function provide full-information feedback.

In this paper, we address the problem of learning from logged bandit feedback. Unlike online learning with bandit feedback, batch learning with bandit feedback does not require interactive experimental control over the system. Furthermore, it enables the reuse of existing data and offline cross-validation techniques for model selection (e.g., "which features to use?", "which learning algorithm to use?", etc.).

To design algorithms for batch learning from bandit feedback, *counterfactual* estimators (Bottou et al., 2013) of a system's performance can be used to estimate how other systems would have performed if they had been in control of choosing predictions. Such estimators have been developed recently for the off-policy evaluation problem (Dudík et al., 2011; Li et al., 2011, 2014), where data collected from the interaction logs of one bandit algorithm is used to evaluate another system.

Our approach to counterfactual learning centers around the insight that, to perform robust learning, it is not sufficient to have just an unbiased estimator of the off-policy system's performance. We must also reason about how the variances of these estimators differ across the hypothesis space, and pick the hypothesis that has the best possible guarantee (tightest conservative bound) for its performance. We first prove generalization error bounds for a *stochastic hypothesis* family using an empirical Bernstein argument (Maurer and Pontil, 2009). This builds on recent approaches to deriving confidence intervals for counterfactual estimators (Bottou et al., 2013; Thomas et al., 2015). By relating the generalization error to the empirical sample variance of different hypotheses, we can effectively penalize the hypotheses with large variance during training using a data-dependant regularizer. In analogy to Structural Risk Minimization for full-information feedback (Wapnik and Tscherwonenkis, 1979), the constructive nature of these bounds suggests a general principle—Counterfactual Risk Minimization (CRM)—for designing methods for batch learning from bandit feedback.

Using the CRM principle, we derive a new learning algorithm—Policy Optimizer for Exponential Models (POEM)—for structured output prediction. The training objective is decomposed using repeated variance linearization, and optimizing it using AdaGrad (Duchi et al., 2011) yields a fast and effective algorithm. We evaluate POEM on several multi-label classification problems, verify that its empirical performance supports the theory, and demonstrates substantial gain in generalization performance over the state-of-the-art.

This paper is an extended version of Swaminathan and Joachims (2015), adding the following contributions. First, it provides the proof of the main generalization error bound upon which the CRM principle is based. Second, it derives and details the Iterated Variance Majorization Algorithm for training POEM, which was only sketched in Swaminathan and Joachims (2015). Third, the paper provides a first real-world experiment using POEM for learning a high precision classifier for information retrieval using logged click data.

The remainder of this paper is structured as follows. We review existing approaches in Section 2. The learning setting is detailed in Section 3, and contrasted with supervised learning. In Section 4, we derive the Counterfactual Risk Minimization learning principle and provide a rule of thumb for setting hyper-parameters. In Section 5, we instantiate the CRM principle for structured output prediction using exponential models and construct an

efficient decomposition of the objective for stochastic optimization. Empirical evaluations are reported in Section 6 and a real-world application is described in Section 7. We conclude with future directions and discussion in Section 8.

2. Related Work

Existing approaches for batch learning from logged bandit feedback fall into two categories. The first approach is to reduce the problem to supervised learning. In principle, since the logs give us an incomplete view of the feedback for different predictions, one could first use regression to estimate a feedback oracle for unseen predictions, and then use any supervised learning algorithm using this feedback oracle. Such a two-stage approach is known to not generalize well (Beygelzimer and Langford, 2009). More sophisticated techniques using the Offset Tree algorithm (Beygelzimer and Langford, 2009) allow us to perform batch learning when the space of possible predictions is small. In contrast, our approach generalizes structured output prediction, with exponential-sized prediction spaces. In particular, we apply our approach to multilabel classification problems. When the number of labels is K, the number of possible predictions is 2^K. A direct application of the Offset tree algorithm requires $\mathcal{O}(2^K)$ space and only guarantees regret $\mathcal{O}((2^K - 1)r)$ where r is the regret of the underlying binary classifier. Our approach directly tackles the problem using popular models from structured prediction instead, using computation and space complexity that mimics supervised approaches to the problem.

The second approach to batch learning from bandit feedback uses propensity scoring (Rosenbaum and Rubin, 1983) to derive unbiased estimators from the interaction logs (Bottou et al., 2013). These estimators are used for a small set of candidate policies, and the best estimated candidate is picked via exhaustive search. In contrast, our approach can be optimized via gradient descent, over hypothesis families (of infinite size) that are equally as expressive as those used in supervised learning. In particular, we build on recent work that develops confidence bounds for counterfactual estimators (Bottou et al., 2013; Thomas et al., 2015) using empirical Bernstein bounds. Our key insight is that these confidence intervals are not merely observable but can be efficiently optimized during training. Other recent bounds derived from analyzing Renyi divergences (Cortes et al., 2010) can analogously be co-opted in our approach to counterfactual learning.

Our approach builds on counterfactual estimators that have been developed for off-policy evaluation. The inverse propensity scoring approach can work well when we have a good model of the historical algorithm (Strehl et al., 2010; Li et al., 2014, 2015), and doubly robust estimators (Dudík et al., 2011) are even more effective when we additionally have a good model of the feedback. In our work, we focus on the inverse propensity scoring estimator, but the results we derive hold equally for the doubly robust estimators.

In the current work, we concentrate on the case where the historical algorithm was a stationary, stochastic policy. Techniques like exploration scavenging (Langford et al., 2008) and bootstrapping (Mary et al., 2014) allow us to perform counterfactual evaluation even when the historical algorithm was deterministic or adaptive.

Our strategy of picking the hypothesis with the tightest conservative bound on performance mimics similar successful approaches in other problems like supervised learning (Wapnik and Tscherwonenkis, 1979), risk averse multi-armed bandits (Galichet et al., 2013),

regret minimizing contextual bandits (Langford and Zhang, 2008) and reinforcement learning (Garcia and Fernandez, 2012). Beyond the problem of batch learning from bandit feedback, our approach can have implications for several applications that require learning from logged bandit feedback data: warm-starting multi-armed bandits (Shivaswamy and Joachims, 2012) and contextual bandits (Strehl et al., 2010), pre-selecting retrieval functions for search engines (Hofmann et al., 2013), policy evaluation for contextual bandits (Li et al., 2011), and reinforcement learning (Thomas et al., 2015) to name a few.

3. Learning Setting: Batch Learning with Logged Bandit Feedback

Consider a structured output prediction problem that takes as input $x \in \mathcal{X}$ and outputs a prediction $y \in \mathcal{Y}$. For example, in multi-label document classification, x could be a news article and y a bitvector indicating the labels assigned to this article. The inputs are assumed drawn from a fixed but unknown distribution $\Pr(\mathcal{X})$, $x \overset{i.i.d.}{\sim} \Pr(\mathcal{X})$. Consider a hypothesis space $\mathcal{H}$ of *stochastic policies*. A hypothesis $h(\mathcal{Y} \mid x) \in \mathcal{H}$ defines a probability distribution over the output space $\mathcal{Y}$, and the hypothesis makes predictions by *sampling*, $y \sim h(\mathcal{Y} \mid x)$. Note that this definition also includes deterministic hypotheses, where the distributions assign probability 1 to a single y. For notational convenience, denote $h(\mathcal{Y} \mid x)$ by $h(x)$, and the probability assigned by $h(x)$ to y as $h(y \mid x)$. We will abuse notation slightly and use $(x, y) \sim h$ to refer to samples drawn from the joint distribution, $x \sim \Pr(\mathcal{X}), y \sim h(\mathcal{Y} \mid x)$. When it is clear from the context, we will drop $(x, y) \sim h$ and simply write h.

In interactive learning systems, we only observe feedback $\delta(x, y)$ for the y sampled from $h(x)$. In this work, feedback $\delta : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}$ is a cardinal loss that is only observed at the sampled data points. Small values for $\delta(x, y)$ indicate user satisfaction with y for x, while large values indicate dissatisfaction. The expected loss—called risk—of a hypothesis $R(h)$ is defined as,

$$R(h) = \mathbb{E}_{x \sim \Pr(\mathcal{X})} \mathbb{E}_{y \sim h(x)} \left[\delta(x, y) \right] = \mathbb{E}_h \left[\delta(x, y) \right].$$

The goal of the system is to minimize risk, or equivalently, maximize expected user satisfaction. The aim of learning is to find a hypothesis $h \in \mathcal{H}$ that has minimum risk.

We wish to re-use the interaction logs of these systems for batch learning. Assume that its historical algorithm acted according to a *stationary* policy $h_0(x)$ (also called logging policy). The data collected from this system is

$$\mathcal{D} = \{(x_1, y_1, \delta_1), \ldots, (x_n, y_n, \delta_n)\},$$

where $y_i \sim h_0(x_i)$ and $\delta_i \equiv \delta(x_i, y_i)$.

Sampling bias. $\mathcal{D}$ cannot be used to estimate $R(h)$ for a new hypothesis h using the estimator typically used in supervised learning. We ideally need either full information about $\delta(x_i, \cdot)$ or need samples $y \sim h(x_i)$ to directly estimate $R(h)$. This explains why, in practice, model selection over a small set of candidate systems is typically done via A/B tests, where the candidates are deployed to collect new data sampled according to $y \sim h(x)$ for each hypothesis h. A relative comparison of the assumptions, hypotheses, and principles used in supervised learning vs. our learning setting is outlined in Table 1. Fundamentally, batch learning with bandit feedback is hard because $\mathcal{D}$ is both *biased* (predictions favored by the historical algorithm will be over-represented) and *incomplete* (feedback for other predictions will not be available) for learning.

	Supervised	Batch w/bandit
Distribution	$(x, y^*) \sim \Pr(\mathcal{X} \times \mathcal{Y})$	$x \sim \Pr(\mathcal{X}), y \sim h_0(x)$
Data $\mathcal{D}$	$\{x_i, y_i^*\}$	$\{x_i, y_i, \delta_i, p_i\}$
Hypothesis h	$y = h(x)$	$y \sim h(\mathcal{Y} \mid x)$
Loss	$\Delta(y^*, \cdot)$ known	$\delta(x, \cdot)$ unknown
Objective: argmin_h	$\hat{R}(h) + C \cdot Reg(\mathcal{H})$	$\hat{R}^M(h) + C \cdot Reg(\mathcal{H}) + \lambda \cdot \sqrt{\frac{\boldsymbol{Var}(h)}{n}}$

Table 1: Comparison of assumptions, hypotheses and learning principles for supervised learning and batch learning with bandit feedback.

4. Learning Principle: Counterfactual Risk Minimization

The distribution mismatch between h_0 and any hypothesis $h \in \mathcal{H}$ can be addressed using importance sampling, which corrects the sampling bias as:

$$R(h) = \mathbb{E}_h\left[\delta(x, y)\right] = \mathbb{E}_{h_0}\left[\delta(x, y)\frac{h(y \mid x)}{h_0(y \mid x)}\right].$$

This motivates the propensity scoring approach of Rosenbaum and Rubin (1983). During the operation of the logging policy, we keep track of the propensity, $h_0(y \mid x)$ of the historical system to generate y for x. From these propensity-augmented logs

$$\mathcal{D} = \{(x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n)\},$$

where $p_i \equiv h_0(y_i \mid x_i)$, we can derive an unbiased estimate of $R(h)$ via Monte Carlo approximation,

$$\hat{R}(h) = \frac{1}{n}\sum_{i=1}^{n}\delta_i\frac{h(y_i \mid x_i)}{p_i}. \tag{1}$$

At first thought, one may think that directly estimating $\hat{R}(h)$ over $h \in \mathcal{H}$ and picking the empirical minimizer is a valid learning strategy. Unfortunately, there are several pitfalls.

First, this strategy is not invariant to additive transformations of the loss and will give degenerate results if the loss is not appropriately scaled. In Section 4.3, we develop intuition for why this is so, and derive the optimal scaling of δ. For now, assume that $\forall x, \forall y, \delta(x, y) \in [-1, 0]$.

Second, this estimator has unbounded variance, since $p_i \simeq 0$ in $\mathcal{D}$ can cause $\hat{R}(h)$ to be arbitrarily far away from the true risk $R(h)$. This can be fixed by "clipping" the importance sampling weights (Ionides, 2008; Strehl et al., 2010; Bottou et al., 2013; Cortes et al., 2010)

$$R^M(h) = \mathbb{E}_{h_0}\left[\delta(x, y)\min\left\{M, \frac{h(y \mid x)}{h_0(y \mid x)}\right\}\right],$$
$$\hat{R}^M(h) = \frac{1}{n}\sum_{i=1}^{n}\delta_i\min\left\{M, \frac{h(y_i \mid x_i)}{p_i}\right\}.$$

$M \geq 1$ is a hyper-parameter chosen to trade-off bias and variance in the estimate, where smaller values of M induce larger bias in the estimate. Optimizing $\hat{R}^M(h)$ through exhaustive enumeration over $\mathcal{H}$ yields the Inverse Propensity Scoring (IPS) training objective

$$\hat{h}^{IPS} = \operatorname*{argmin}_{h \in \mathcal{H}} \left\{ \hat{R}^M(h) \right\}. \tag{2}$$

This objective captures the essence of previous offline policy optimization approaches (Bottou et al., 2013; Strehl et al., 2010). These approaches differ from Equation (2) in the specific way the importance sampling weights are clipped, and frame the optimization problem as a maximization of counterfactual rewards as opposed to minimization of counterfactual risk.

Third, importance sampling typically estimates $\hat{R}^M(h)$ of different hypotheses $h \in \mathcal{H}$ with vastly different variances. Consider two hypotheses h_1 and h_2, where h_1 is similar to h_0, but where h_2 samples predictions that were not well explored by h_0. Importance sampling gives us low-variance estimates for $\hat{R}^M(h_1)$, but highly variable estimates for $\hat{R}^M(h_2)$. Intuitively, if we can develop variance-sensitive confidence bounds over the hypothesis space, optimizing a conservative confidence bound should find a h whose $R(h)$ will not be much worse, with high probability.

4.1 Generalization Error Bound

A standard analysis would give a bound that is agnostic to the variance introduced by importance sampling. Following our intuition above, we derive a higher order bound that includes the variance term using empirical Bernstein bounds (Maurer and Pontil, 2009). To develop such a generalization error bound, we first need a concept of capacity for stochastic hypothesis classes. Our strategy is to define an auxiliary deterministic function class $\mathcal{F}_{\mathcal{H}}$ for $\mathcal{H}$ and directly use covering numbers for $\mathcal{F}_{\mathcal{H}}$ conditioned on a sample $\mathcal{D}$. We start by defining the auxiliary deterministic function class $\mathcal{F}_{\mathcal{H}}$.

Definition 1 *For any stochastic class $\mathcal{H}$, define an auxiliary function class $\mathcal{F}_{\mathcal{H}} = \{f_h : \mathcal{X} \times \mathcal{Y} \mapsto [0, 1]\}$. Each $h \in \mathcal{H}$ corresponds to a function $f_h \in \mathcal{F}_{\mathcal{H}}$,*

$$f_h(x, y) = 1 + \frac{\delta(x, y)}{M} \min \left\{ M, \frac{h(y \mid x)}{h_0(y \mid x)} \right\}. \tag{3}$$

Based on this auxiliarity function class $\mathcal{F}_{\mathcal{H}}$, we will study the convergence of $\hat{R}^M(h) \rightarrow R^M(h)$. A key insight is the following relationship between h and f_h.

Lemma 2 *For any stochastic hypothesis h, the clipped risk $R^M(h)$ and the expected value of f_h under the data generating distribution are related as*

$$\mathbb{E}_{h_0} \left[f_h(x, y) \right] = 1 + \frac{R^M(h)}{M}. \tag{4}$$

Proof Note that f_h is a deterministic and bounded function. From the definition of f_h and by linearity of expectation,

$$\begin{aligned}\mathbb{E}_{h_0}\left[f_h(x,y)\right] &= \mathbb{E}_{h_0}\left[1+\frac{\delta(x,y)}{M}\min\left\{M,\frac{h(y\mid x)}{h_0(y\mid x)}\right\}\right]\\ &= 1+\frac{1}{M}\mathbb{E}_{h_0}\left[\delta(x,y)\min\left\{M,\frac{h(y\mid x)}{h_0(y\mid x)}\right\}\right]\\ &= 1+\frac{R^M(h)}{M}\end{aligned}$$

■

As a consequence of Lemma 2, we can use classic notions of capacity for $\mathcal{F}_\mathcal{H}$ to reason about the convergence of $\hat{R}^M(h) \to R^M(h)$. Recall the covering number $\mathcal{N}_\infty(\epsilon,\mathcal{F},n)$ for a function class $\mathcal{F}$.[1] Define an $\epsilon-$cover $\mathcal{N}(\epsilon,A,\|\cdot\|_\infty)$ for a set $A\subseteq\mathbb{R}^n$ to be the size of the smallest cardinality subset $A_0\subseteq A$ such that A is contained in the union of balls of radius ϵ centered at points in A_0, in the metric induced by $\|\cdot\|_\infty$. The covering number is,

$$\mathcal{N}_\infty(\epsilon,\mathcal{F},n) = \sup_{(x_i,y_i)\in(\mathcal{X}\times\mathcal{Y})^n}\mathcal{N}(\epsilon,\mathcal{F}(\{(x_i,y_i)\}),\|\cdot\|_\infty),$$

where $\mathcal{F}(\{(x_i,y_i)\})$ is the function class conditioned on sample $\{(x_i,y_i)\}$,

$$\mathcal{F}(\{(x_i,y_i)\}) = \{(f(x_1,y_1),\ldots,f(x_n,y_n)) : f\in\mathcal{F}\}.$$

Our measure for the capacity of our stochastic class $\mathcal{H}$ to "fit" a sample of size n shall be $\mathcal{N}_\infty(\frac{1}{n},\mathcal{F}_\mathcal{H},2n)$.

For a compact notation, define the random variable $u_h \equiv \delta(x,y)\min\left\{M,\frac{h(y|x)}{h_0(y|x)}\right\}$ with mean $\overline{u}_h = R^M(h)$. The sample $\mathcal{D}$ contains n i.i.d. random variables ${u_h}^i \equiv \delta_i\min\{M,\frac{h(y_i|x_i)}{p_i}\}$. Define the sample mean and variance of ${u_h}^i$

$$\hat{\overline{u}}_h \equiv \frac{1}{n}\sum_{i=1}^n {u_h}^i = \hat{R}^M(h),$$

$$\hat{\boldsymbol{Var}}(u_h) \equiv \frac{1}{n-1}\sum_{i=1}^n({u_h}^i - \hat{\overline{u}}_h)^2.$$

Theorem 3 *With probability at least $1-\gamma$ in the random vector $(x_1,y_1)\cdots(x_n,y_n) \overset{i.i.d.}{\sim} h_0$, with observed losses $\delta_1,\ldots,\delta_n$, for $n\geq 16$ and a stochastic hypothesis space $\mathcal{H}$ with capacity $\mathcal{N}_\infty(\frac{1}{n},\mathcal{F}_\mathcal{H},2n)$,*

$$\forall h\in\mathcal{H}:\quad R(h)\leq\hat{R}^M(h)+\sqrt{18\frac{\hat{\boldsymbol{Var}}(u_h)\mathcal{Q}_\mathcal{H}(n,\gamma)}{n}}+M\frac{15\mathcal{Q}_\mathcal{H}(n,\gamma)}{n-1},$$

$$\textit{where, } \mathcal{Q}_\mathcal{H}(n,\gamma)\equiv\log(\frac{10\cdot\mathcal{N}_\infty(\frac{1}{n},\mathcal{F}_\mathcal{H},2n)}{\gamma}),\quad 0<\gamma<1.$$

1. Refer Anthony and Bartlett (2009); Maurer and Pontil (2009) and the references therein for a comprehensive treatment of covering numbers.

Proof The proof follows from Theorem 6 of Maurer and Pontil (2009) applied to the deterministic function class $\mathcal{F}_{\mathcal{H}}$. We sketch the main argument using symmetrization and Rademacher variables here.

Define the random variable $s_h = 1 + \frac{u_h}{M}$ with mean $\mathbb{E}_{h_0}[s_h]$ and variance $\boldsymbol{Var}(s_h)$. Observe that $\mathbb{E}_{h_0}[s_h] = 1 + \frac{R^M(h)}{M}$ from Lemma 2. Let ${s_h}^i = 1 + \frac{{u_h}^i}{M}$. The sample $\mathcal{D}$ essentially contains n i.i.d. observations of s_h. Let $\hat{\bar{s}}_h$ and $\hat{\boldsymbol{Var}}(s_h)$ denote the empirical mean and variance of $\{{s_h}^i\}_{i=1}^n$ respectively. Observe that $\hat{\boldsymbol{Var}}(s_h) = \frac{\hat{\boldsymbol{Var}}(u_h)}{M^2}$. Abusing notation slightly, we will use boldface $\boldsymbol{s_h}$ to refer to the sample $\{{s_h}^i\}_{i=1}^n$.

We begin with Bennet's inequality.

For $s, \{s^i\}_{i=1}^n$ i.i.d. bounded random variables in $[0,1]$ having mean $\mathbb{E}[s]$ and variance $\boldsymbol{Var}(s)$, with probability at least $1-\gamma$ in $\{s^i\}_{i=1}^n \equiv \boldsymbol{s}$,

$$\mathbb{E}[s] - \hat{\bar{s}} \leq \sqrt{\frac{2\boldsymbol{Var}(s)\log 1/\gamma}{n}} + \frac{\log 1/\gamma}{3n}. \tag{5}$$

Intuitively, Bennet's inequality tells us that the estimate $\hat{\bar{s}}$ has lower accuracy if $\boldsymbol{Var}(s)$ is high, which exactly captures our intuition about the variance introduced by importance sampling when estimating the risk of a hypothesis "far" from h_0. However, the diameter of this confidence interval depends on the unobservable $\boldsymbol{Var}(s)$.

We recite Theorem 11 from Maurer and Pontil (2009) that gives a variance-sensitive bound with an observable confidence interval, which they call an Empirical Bernstein bound.

Under the same conditions as Bennet's inequality (5), let $n \geq 2, \hat{\boldsymbol{Var}}(s)$ represent the empirical variance of $\{s^i\}_{i=1}^n$. With probability at least $1-\gamma$,

$$\mathbb{E}[s] - \hat{\bar{s}} \leq \sqrt{\frac{2\hat{\boldsymbol{Var}}(s)\log 2/\gamma}{n}} + \frac{7\log 2/\gamma}{3(n-1)}. \tag{6}$$

This follows from confidence bounds on the sample standard deviation $\sqrt{\hat{\boldsymbol{Var}}(s)}$ compared to the true standard deviation $\mathbb{E}_{\boldsymbol{s}}\left[\sqrt{\hat{\boldsymbol{Var}}(s)}\right]$. Based on this bound, Maurer and Pontil (2009) define two Lipschitz continuous functions, $\Phi, \Psi : [0,1]^n \times \mathbb{R}_+ \to \mathbb{R}$.

$$\Phi(\boldsymbol{s}, t) = \hat{\bar{s}} + \sqrt{\frac{2\hat{\boldsymbol{Var}}(s)t}{n}} + \frac{7t}{3(n-1)}$$
$$\Psi(\boldsymbol{s}, t) = \hat{\bar{s}} + \sqrt{\frac{18\hat{\boldsymbol{Var}}(s)t}{n}} + \frac{11t}{n-1}.$$

These functions are Lipschitz continuous,

$$\Phi(\boldsymbol{s}, t) - \Phi(\boldsymbol{s'}, t) \leq (1 + 2\sqrt{\frac{t}{n}})\|\boldsymbol{s} - \boldsymbol{s'}\|_\infty$$
$$\Psi(\boldsymbol{s}, t) - \Psi(\boldsymbol{s'}, t) \leq (1 + 6\sqrt{\frac{t}{n}})\|\boldsymbol{s} - \boldsymbol{s'}\|_\infty. \tag{7}$$

The inequalities follow directly from $\sqrt{\hat{\boldsymbol{Var}}(s)} - \sqrt{\hat{\boldsymbol{Var}}(s')} \leq \sqrt{2}\|\boldsymbol{s} - \boldsymbol{s'}\|_\infty$.

For the symmetrization argument, consider two sets of n samples $\mathcal{D}$ and $\mathcal{D}'$ drawn from h_0 according to the conditions of Theorem 3 and used to estimate risk of a hypothesis h. This gives rise to two sets of n i.i.d. random variables $\boldsymbol{s}_h$ and $\boldsymbol{s}'_h$. Also define the Rademacher variables $\sigma_1, \dots \sigma_n \overset{i.i.d}{\sim} \mathcal{U}\{-1, 1\}$. Define $(\boldsymbol{\sigma}, \boldsymbol{s}_h, \boldsymbol{s}'_h)$ as the vector who's i^{th} co-ordinate is set to ${s_h}^i$ or ${s'_h}^i$ as specified by σ_i.

$$(\boldsymbol{\sigma}, \boldsymbol{s}_h, \boldsymbol{s}'_h)_i = \begin{cases} {s_h}^i & \text{if } \sigma_i = 1 \\ {s'_h}^i & \text{if } \sigma_i = -1. \end{cases}$$

For a fixed $h \in \mathcal{H}$ and a fixed double sample $\boldsymbol{s}_h, \boldsymbol{s}'_h$ as described above,

$$\Pr_{\sigma}\left[\Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t) \geq \Psi((\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t)\right] \leq 5e^{-t}. \tag{8}$$

This is simply a restatement of Lemma 14 from Maurer and Pontil (2009) and follows by decomposing the event $[\Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t) \geq \Psi((\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t)]$ as $[\Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t) \geq A] \wedge [A \geq \Psi((\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t)]$ where A uses the true mean and variance of s_h. The probability of the first event can be bounded using Bennet's inequality (5), while the second event can be bounded using the empirical Bernstein bound (6) and the confidence bounds on the sample standard deviation $\sqrt{\hat{\boldsymbol{Var}}(s)}$.

Set $t = \log \frac{2}{\gamma}$ and consider $t \geq \log 4$ (i.e. $\gamma \leq \frac{1}{2}$). Equation (6) implies, for any $h \in \mathcal{H}$,

$$\Pr(\Phi(\boldsymbol{s}_h, t) \geq \mathbb{E}[s_h]) \geq \frac{1}{2}. \tag{9}$$

Hence, for any $\rho > 0$,

$$\begin{aligned}
&\Pr_{\mathcal{D}}(\exists h \in \mathcal{H} : \mathbb{E}[s_h] > \Psi(\boldsymbol{s}_h, t) + \rho) = \mathbb{E}_{\mathcal{D}}\left[\sup_{h \in \mathcal{H}} \mathbb{I}\{\mathbb{E}[s_h] > \Psi(\boldsymbol{s}_h, t) + \rho\}\right] \\
&\leq \mathbb{E}_{\mathcal{D}}\left[\sup_{h \in \mathcal{H}} \mathbb{I}\{\mathbb{E}[s_h] > \Psi(\boldsymbol{s}_h, t) + \rho\}\right] 2\Pr(\Phi(\boldsymbol{s}'_h, t) \geq \mathbb{E}\left[s'_h\right]) && \text{Equation (9)} \\
&= 2\mathbb{E}_{\mathcal{D}}\left[\sup_{h \in \mathcal{H}} \mathbb{E}_{\mathcal{D}'}\left[\mathbb{I}\{\mathbb{E}[s_h] > \Psi(\boldsymbol{s}_h, t) + \rho \wedge \Phi(\boldsymbol{s}'_h, t) \geq \mathbb{E}[s_h]\}\right]\right] && \text{since } \mathbb{E}[s_h] = \mathbb{E}\left[s'_h\right] \\
&\leq 2\mathbb{E}_{\mathcal{D}}\mathbb{E}_{\mathcal{D}'}\left[\sup_{h \in \mathcal{H}} \mathbb{I}\{\mathbb{E}[s_h] > \Psi(\boldsymbol{s}_h, t) + \rho \wedge \Phi(\boldsymbol{s}'_h, t) \geq \mathbb{E}[s_h]\}\right] \\
&\leq 2\mathbb{E}_{\mathcal{D}}\mathbb{E}_{\mathcal{D}'}\left[\sup_{h \in \mathcal{H}} \mathbb{I}\{\Phi(\boldsymbol{s}'_h, t) > \Psi(\boldsymbol{s}_h, t) + \rho\}\right] \\
&= 2\mathbb{E}_{\sigma}\mathbb{E}_{\mathcal{D}}\mathbb{E}_{\mathcal{D}'}\left[\sup_{h \in \mathcal{H}} \mathbb{I}\{\Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t) > \Psi((-\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t) + \rho\}\right] && \text{since } \boldsymbol{s}_h, \boldsymbol{s}'_h \text{ are i.i.d.} \\
&\leq 2 \sup_{\mathcal{D}, \mathcal{D}'} \mathbb{E}_{\sigma}\left[\sup_{h \in \mathcal{H}} \mathbb{I}\{\Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t) > \Psi((-\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t) + \rho\}\right] \\
&= 2 \sup_{\mathcal{D}, \mathcal{D}'} \Pr_{\sigma}(\exists h \in \mathcal{H} : \Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t) > \Psi((-\sigma, \boldsymbol{s}_h, \boldsymbol{s}'_h), t) + \rho).
\end{aligned}$$

For a fixed $\mathcal{D}, \mathcal{D}'$, consider the $\epsilon-$cover of $\mathcal{F}_{\mathcal{H}}$, ${\mathcal{F}_{\mathcal{H}}}^0$. Denote the set of stochastic policies that correspond to each $f_h \in {\mathcal{F}_{\mathcal{H}}}^0$ by $\mathcal{H}^0$. We know that $\left|\mathcal{H}^0\right| \leq \mathcal{N}_{\infty}(\epsilon, \mathcal{F}_{\mathcal{H}}, 2n)$ (by

definition of the covering number, and since there is a one-to-one mapping from h to f_h) and $\forall h \in \mathcal{H}, \quad \exists h' \in \mathcal{H}^0$ such that $\|\boldsymbol{s}_h - \boldsymbol{s}_{h'}\|_\infty \leq \epsilon$ and $\|\boldsymbol{s'}_h - \boldsymbol{s'}_{h'}\|_\infty \leq \epsilon$ (by definition of $\epsilon-$cover). Instantiate $\rho = \epsilon(2 + 8\sqrt{\frac{t}{n}})$ and suppose $\exists h \in \mathcal{H}$ such that $\Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s'}_h), t) > \Psi((-\sigma, \boldsymbol{s}_h, \boldsymbol{s'}_h), t) + \rho$. Since Φ and Ψ are Lipschitz continuous, as demonstrated in Equation (7), hence there must exist a $h' \in \mathcal{H}^0$ such that $\Phi((\sigma, \boldsymbol{s}_{h'}, \boldsymbol{s'}_{h'}), t) > \Psi((-\sigma, \boldsymbol{s}_{h'}, \boldsymbol{s'}_{h'}), t)$. Hence,

$$
\begin{aligned}
&\Pr_\sigma(\exists h \in \mathcal{H} : \Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s'}_h), t) > \Psi((-\sigma, \boldsymbol{s}_h, \boldsymbol{s'}_h), t) + \epsilon(2 + 8\sqrt{\frac{t}{n}})) \\
&\leq \Pr_\sigma(\exists h \in \mathcal{H}^0 : \Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s'}_h), t) > \Psi((-\sigma, \boldsymbol{s}_h, \boldsymbol{s'}_h), t)) \\
&\leq \sum_{h \in \mathcal{H}^0} \Pr_\sigma(\Phi((\sigma, \boldsymbol{s}_h, \boldsymbol{s'}_h), t) > \Psi((-\sigma, \boldsymbol{s}_h, \boldsymbol{s'}_h), t)) \\
&\leq 5e^{-t}\mathcal{N}_\infty(\epsilon, \mathcal{F}_\mathcal{H}, 2n) \qquad \text{Equation (8) .}
\end{aligned}
$$

In short,

$$
\Pr_\mathcal{D}(\exists h \in \mathcal{H} : \mathbb{E}[s_h] > \Psi(\boldsymbol{s}_h, t) + \epsilon(2 + 8\sqrt{\frac{t}{n}})) \leq 10e^{-t}\mathcal{N}_\infty(\epsilon, \mathcal{F}_\mathcal{H}, 2n).
$$

Setting $10e^{-t}\mathcal{N}_\infty(\epsilon, \mathcal{F}_\mathcal{H}, 2n) = \gamma$ we get $t_\gamma = \log \frac{10\mathcal{N}_\infty(\epsilon, \mathcal{F}_\mathcal{H}, 2n)}{\gamma} > 1$. Moreover, $\frac{2(t_\gamma+1)}{n} \leq \frac{2(t_\gamma+1)}{n-1} \leq \frac{4t_\gamma}{n-1}$ and for $n \geq 16$, $8\sqrt{\frac{t_\gamma}{n}} \leq 2t_\gamma$. Substituting $\epsilon = \frac{1}{n}$ and simplifying,

$$
\Pr_\mathcal{D}(\exists h \in \mathcal{H} : \mathbb{E}[s_h] > \hat{\bar{s}}_h + \sqrt{\frac{18\hat{\boldsymbol{Var}}(s_h)t_\gamma}{n}} + \frac{15t_\gamma}{n-1}) \leq \gamma.
$$

Finally, $\mathbb{E}[s_h] = 1 + \frac{R^M(h)}{M}, \hat{\bar{s}}_h = 1 + \frac{\hat{R}^M(h)}{M}$ and $\hat{\boldsymbol{Var}}(s_h) = \frac{\hat{\boldsymbol{Var}}(u_h)}{M^2}$. Since $\delta(\cdot, \cdot) \leq 0$, hence $R(h) \leq R^M(h)$. Putting it all together,

$$
\Pr_\mathcal{D}(\exists h \in \mathcal{H} : R(h) > \hat{R}^M(h) + \sqrt{\frac{18\hat{\boldsymbol{Var}}(u_h)t_\gamma}{n}} + \frac{15Mt_\gamma}{n-1}) \leq \gamma.
$$

■

4.2 CRM Principle

The generalization error bound from the previous section is constructive in the sense that it motivates a general principle for designing machine learning methods for batch learning from bandit feedback. In particular, a learning algorithm following this principle should jointly optimize the estimate $\hat{R}^M(h)$ as well as its empirical standard deviation, where the latter serves as a *data-dependent regularizer*.

$$
\hat{h}^{CRM} = \operatorname*{argmin}_{h \in \mathcal{H}} \left\{ \hat{R}^M(h) + \lambda\sqrt{\frac{\hat{\boldsymbol{Var}}(u_h)}{n}} \right\}. \tag{10}
$$

$M \geq 1$ and $\lambda \geq 0$ are regularization hyper-parameters. When $\lambda = 0$, we recover the Inverse Propensity Scoring objective of Equation (2). In analogy to Structural Risk Minimization (Wapnik and Tscherwonenkis, 1979), we call this principle *Counterfactual Risk Minimization*, since both pick the hypothesis with the tightest upper bound on the true risk $R(h)$.

4.3 Optimal loss scaling

When performing supervised learning with true labels y^* and a loss function $\Delta(y^*, \cdot)$, empirical risk minimization using the standard estimator is invariant to additive translation and multiplicative scaling of Δ. The risk estimators $\hat{R}(h)$ and $\hat{R}^M(h)$ in bandit learning, however, crucially require $\delta(\cdot, \cdot) \in [-1, 0]$.

Consider, for example, the case of $\delta(\cdot, \cdot) \geq 0$. The training objectives in Equation (2) (IPS) and Equation (10) (CRM) become degenerate! A hypothesis $h \in \mathcal{H}$ that completely avoids the sample $\mathcal{D}$ (i.e. $\forall i = 1, \ldots, n, h(y_i \mid x_i) = 0$) trivially achieves the best possible $\hat{R}^M(h)$ $(= 0)$ with 0 empirical variance. This degeneracy arises partially because when $\delta(\cdot, \cdot) \geq 0$, the objectives optimize a *lower* bound on $R(h)$, whereas what we need is an *upper* bound.

For any bounded loss $\delta(\cdot, \cdot) \in [\bigtriangledown, \bigtriangleup]$, we have, $\forall x$

$$\mathbb{E}_{y \sim h(x)} \left[\delta(x, y)\right] \leq \bigtriangleup + \mathbb{E}_{y \sim h_0(x)} \left[(\delta(x, y) - \bigtriangleup) \min \left\{ M, \frac{h(y \mid x)}{h_0(y \mid x)} \right\}\right].$$

Since the optimization objectives in Equations (2),(10) are unaffected by a constant scale factor (e.g., $\bigtriangleup - \bigtriangledown$), we should transform $\delta \mapsto \delta'$ to derive a conservative training objective,

$$\delta' \equiv \{\delta - \bigtriangleup\} / \{\bigtriangleup - \bigtriangledown\}.$$

Such a transformation captures the following assumption: for an input $x \in \mathcal{D}$, if a new hypothesis $h \neq h_0$ samples an unexplored y not seen in $\mathcal{D}$, in the worst case it will incur a loss of $\bigtriangleup$. This is clearly a very conservative assumption, and we foresee future work that relaxes this using additional assumptions about $\delta(\cdot, \cdot)$ and $\mathcal{Y}$.

4.4 Selecting hyper-parameters

We propose selecting the hyper-parameters $M \geq 1$ and $\lambda \geq 0$ via cross validation. However, we must be careful not to set M too small or λ too big. The estimated risk $\hat{R}^M(h) \in [-M, 0]$, while the variance penalty $\sqrt{\frac{\hat{\boldsymbol{Var}}(u_h)}{n}} \in \left[0, \frac{M}{2\sqrt{n}}\right]$. If M is too small, all the importance sampling weights will be clipped and all hypotheses will have the same biased estimate of risk $M\hat{R}^M(h_0)$. Similarly, if $\lambda \gg 0$, a hypothesis $h \in \mathcal{H}$ that completely avoids $\mathcal{D}$ (i.e. $\forall i = 1, \ldots, n, h(y_i \mid x_i) = 0$) has $\hat{R}^M(h)$ $(= 0)$ with 0 empirical variance. So, it will achieve the best possible training objective of 0. As a rule of thumb, we can calibrate M and λ so that the estimator is unbiased and the objective is negative for some $h \in \mathcal{H}$. When $h_0 \in \mathcal{H}$, $M \simeq \max\{p_i\} / \min\{p_i\}$ and $\left\{ \hat{R}^M(h_0) + \lambda \sqrt{\frac{\hat{\boldsymbol{Var}}(u_{h_0})}{n}} \right\} < 0$ are natural choices.

4.5 When is counterfactual learning possible?

The bounds in Theorem 3 are with respect to the randomness in h_0. Known impossibility results for counterfactual evaluation using h_0 (Langford et al., 2008) also apply to counterfactual learning. In particular, if h_0 was deterministic, or even stochastic but without full support over $\mathcal{Y}$, it is easy to engineer examples involving the unexplored $y \in \mathcal{Y}$ that guarantee sub-optimal learning even as $|\mathcal{D}| \to \infty$. Similarly, lower bounds for learning under covariate shift (Cortes et al., 2010) also apply to counterfactual learning. Finally, a stochastic h_0 with heavier tails need not always allow more effective learning. From importance sampling theory (Owen, 2013), what really matters is how well h_0 explores the regions of $\mathcal{Y}$ with favorable losses.

5. Learning Algorithm: POEM

We now use the CRM principle to derive an efficient algorithm for structured output prediction using linear rules. Classic learning methods for structured output prediction based on full-information feedback, e.g. structured support vector machines (Tsochantaridis et al., 2004) and conditional random fields (Lafferty et al., 2001), predict using

$$h_w^{sup}(x) = \operatorname*{argmax}_{y \in \mathcal{Y}} \{w \cdot \phi(x, y)\}, \tag{11}$$

where w is a $d-$dimensional weight vector, and $\phi(x, y)$ is a $d-$dimensional joint feature map. For example, in multi-label document classification, for a news article x and a possible assignment of labels y represented as a bitvector, $\phi(x, y)$ could simply be a concatenation $\overline{x} \otimes y$ of the bag-of-words features of the document ($\overline{x}$), one copy for each of the assigned labels in y. Several efficient inference algorithms have been developed to solve Equation (11).

The POEM algorithm that is derived in this section uses the same parameterization of the hypothesis space as in Equation (11). However, it considers the following expanded class of Stochastic Softmax Rules based on this parameterization, which contains the deterministic rule in Equation (11) as a limiting case.

5.1 Stochastic Softmax Rules

Consider the following stochastic family $\mathcal{H}_{lin}$, parametrized by w. A hypothesis $h_w(x) \in \mathcal{H}_{lin}$ samples y from the distribution

$$h_w(y \mid x) = \exp(w \cdot \phi(x, y))/\mathbb{Z}(x).$$

$\mathbb{Z}(x) = \sum_{y' \in \mathcal{Y}} \exp(w \cdot \phi(x, y'))$ is the partition function. This can be thought of as the "soft-max" variant of the "hard-max" rules from Equation (11). Additionally, for a *temperature* multiplier $\alpha > 1, w \mapsto \alpha w$ induces a more "peaked" distribution $h_{\alpha w}$ that preserves the modes of h_w, and intuitively is a "more deterministic" variant of h_w.

h_w lies in the exponential family of distributions, and has a simple gradient,

$$\nabla h_w(y \mid x) = h_w(y \mid x) \left\{\phi(x, y) - \mathbb{E}_{y' \sim h_w(x)} \left[\phi(x, y')\right]\right\}.$$

5.2 POEM Training Objective

Consider a bandit structured-output data set $\mathcal{D} = \{(x_1, y_1, \delta_1, p_1), \ldots, (x_n, y_n, \delta_n, p_n)\}$. In multi-label document classification, this data could be collected from an interactive labeling system, where each y indicates the labels predicted by the system for a document x. The feedback $\delta(x, y)$ is how many labels (but not which ones) were correct. To perform learning, first we scale the losses as outlined in Section 4.3. Next, instantiating the CRM principle of Equation (10) for $\mathcal{H}_{lin}$, (using notation analogous to that in Theorem 3, adapted for $\mathcal{H}_{lin}$), yields the POEM training objective.

$$w^* = \operatorname*{argmin}_{w \in \mathbb{R}^d} \hat{\overline{u}}_w + \lambda \sqrt{\frac{\hat{\boldsymbol{Var}}(u_w)}{n}}, \tag{12}$$

$$\begin{aligned} \text{with} \qquad & {u_w}^i \equiv \delta_i \min\{M, \frac{\exp(w \cdot \phi(x_i, y_i))}{p_i \cdot \mathbb{Z}(x_i)}\}, \\ & \hat{\overline{u}}_w \equiv \frac{1}{n} \sum_{i=1}^{n} {u_w}^i, \\ & \hat{\boldsymbol{Var}}(u_w) \equiv \frac{1}{n-1} \sum_{i=1}^{n} ({u_w}^i - \hat{\overline{u}}_w)^2. \end{aligned}$$

While the objective in Equation (12) is not convex in w (even for $\lambda = 0$), we find that batch and stochastic gradient descent compute h_w that have good generalization error (e.g., L-BFGS out of the box). The key subroutine that enables us to perform efficient gradient descent is a tractable way to compute ${u_w}^i$ and $\nabla_w({u_w}^i)$—both depend on $\mathbb{Z}(x_i)$.

$${u_w}^i = \delta_i \min\{M, \frac{\exp(w \cdot \phi(x_i, y_i))}{p_i \cdot \mathbb{Z}(x_i)}\} \tag{13}$$

$$\nabla_w({u_w}^i) = \begin{cases} 0 & \text{if } \frac{\exp(w \cdot \phi(x_i, y_i))}{p_i \cdot \mathbb{Z}(x_i)} \geq M \\ \frac{\delta_i}{p_i} {u_w}^i \left\{ \phi(x_i, y_i) - \sum_{y'} \left[\phi(x_i, y') \frac{\exp(w \cdot \phi(x_i, y'))}{\mathbb{Z}(x_i)} \right] \right\} & \text{otherwise.} \end{cases}$$

For the special case when $\phi(x, y) = \overline{x} \otimes y$, where y is a bitvector $\in \{0, 1\}^L$, $\mathbb{Z}(x)$ has a simple decomposition.

$$\exp(w \cdot \phi(x, y)) = \prod_{l=1}^{L} \exp(y_l w_l \cdot x),$$

$$\mathbb{Z}(x) = \prod_{l=1}^{L} (1 + \exp(w_l \cdot x)),$$

where L is the length of the bitvector representation of y. For the general case, several approximation schemes have been developed to handle $\mathbb{Z}(x)$ for supervised training of graphical models and we can directly co-opt these for batch learning under bandit feedback as well.

5.3 POEM Iterated Variance Majorization Algorithm

We could use standard batch gradient descent methods to minimize the POEM training objective. In particular, prior work (Yu et al., 2010; Lewis and Overton, 2013) has established theoretically sound modifications to L-BFGS for non-smooth non-convex optimization. However, the following develops a stochastic method that can be much faster.

At first glance, the POEM training objective in Equation (12), specifically the variance term resists stochastic gradient optimization in the presented form. To remove this obstacle, we now develop a Majorization-Minimization scheme, similar in spirit to recent approaches to multi-class SVMs (van den Burg and Groenen, 2014) that can be shown to converge to a local optimum of the POEM training objective. In particular, we will show how to decompose $\sqrt{\hat{\boldsymbol{Var}}(u_w)}$ as a sum of differentiable functions (e.g., $\sum_i {u_w}^i$ or $\sum_i \{{u_w}^i\}^2$) so that we can optimize the overall training objective at scale using stochastic gradient descent.

Proposition 4 *For any w_0 such that $\hat{\boldsymbol{Var}}(u_{w_0}) > 0$,*

$$\begin{aligned}
\sqrt{\hat{\boldsymbol{Var}}(u_w)} &\leq A_{w_0} \sum_{i=1}^{n} {u_w}^i + B_{w_0} \sum_{i=1}^{n} \{{u_w}^i\}^2 + C_{w_0} \\
&= G(w; w_0). \\
A_{w_0} &\equiv -\frac{\hat{\overline{u}}_{w_0}}{(n-1)\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}}, \\
B_{w_0} &\equiv \frac{1}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}}, \\
C_{w_0} &\equiv \frac{n\{\hat{\overline{u}}_{w_0}\}^2}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}} + \frac{\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}}{2}.
\end{aligned}$$

Proof Consider a first order Taylor approximation of $\sqrt{\hat{\boldsymbol{Var}}(u_w)}$ around w_0. Observe that $\sqrt{\cdot}$ is concave.

$$\begin{aligned}
\sqrt{\hat{\boldsymbol{Var}}(u_w)} &\leq \sqrt{\hat{\boldsymbol{Var}}(u_{w_0})} + \nabla_z \sqrt{z} \mid_{z=\hat{\boldsymbol{Var}}(u_{w_0})} (\hat{\boldsymbol{Var}}(u_w) - \hat{\boldsymbol{Var}}(u_{w_0})) \\
&= \sqrt{\hat{\boldsymbol{Var}}(u_{w_0})} + \frac{\hat{\boldsymbol{Var}}(u_w) - \hat{\boldsymbol{Var}}(u_{w_0})}{2\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}} \\
&= \frac{\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}}{2} + \frac{1}{2\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}} \hat{\boldsymbol{Var}}(u_w) \\
&= \frac{\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}}{2} + \frac{\sum_{i=1}^{n} \{{u_w}^i\}^2}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}} + \frac{-n\{\hat{\overline{u}}_w\}^2}{2(n-1)\sqrt{\hat{\boldsymbol{Var}}(u_{w_0})}}.
\end{aligned}$$

Again Taylor approximate $-\{\hat{\overline{u}_w}\}^2$, noting that $-\{\cdot\}^2$ is concave.

$$\begin{aligned}
-\{\hat{\overline{u}_w}\}^2 &\leq -\{\hat{\overline{u}_{w_0}}\}^2 + \nabla_z(-z^2)\mid_{z=\hat{\overline{u}_{w_0}}} (\hat{\overline{u}_w} - \hat{\overline{u}_{w_0}}) \\
&= -\{\hat{\overline{u}_{w_0}}\}^2 + 2\{\hat{\overline{u}_{w_0}}\}^2 - 2\hat{\overline{u}_{w_0}}\hat{\overline{u}_w} \\
&= \{\hat{\overline{u}_{w_0}}\}^2 - \frac{2\hat{\overline{u}_{w_0}}}{n}\sum_{i=1}^{n} {u_w}^i.
\end{aligned}$$

Substituting above and re-arranging terms, we derive the proposition. ■

Iteratively minimizing $w^{t+1} = \text{argmin}_w G(w; w^t)$ ensures that the sequence of iterates $w^1, \ldots, w^{t+1}$ are successive minimizers of $\sqrt{\hat{\boldsymbol{Var}}(u_w)}$. Hence, during an epoch t, POEM proceeds by sampling uniformly $i \sim \mathcal{D}$, computing ${u_w}^i, \nabla {u_w}^i$ and, for learning rate η, updating

$$w \leftarrow w - \eta\{\nabla {u_w}^i + \lambda\sqrt{n}(A_{w_t}\nabla {u_w}^i + 2B_{w_t}{u_w}^i\nabla {u_w}^i)\}.$$

After each epoch, $w^{t+1} \leftarrow w$, and iterated minimization proceeds until convergence.

The complete algorithm is summarized as Algorithm 1. Software implementing POEM is available at http://www.cs.cornell.edu/~adith/poem/ for download, as is all the code and data needed to run each of the experiments reported in Section 6.

6. Empirical Evaluation

We now empirically evaluate the prediction performance and computational efficiency of POEM on a broad range of scenarios. To be able to control these experiments effectively, we derive bandit feedback from existing full-information data sets. As the learning task, we consider multi-label classification with input $x \in \mathbb{R}^p$ and prediction $y \in \{0,1\}^q$. Popular supervised algorithms that solve this problem include Structured SVMs (Tsochantaridis et al., 2004) and Conditional Random Fields (Lafferty et al., 2001). In the simplest case, CRF essentially performs logistic regression for each of the q labels independently. As outlined in Section 5, we use a joint feature map: $\phi(x, y) = x \otimes y$. We conducted experiments on different multi-label data sets collected from the LibSVM repository, with different ranges for p (features), q (labels) and n (samples) represented as summarized in Table 2.

Experiment methodology. We employ the Supervised $\mapsto$ Bandit conversion (Beygelzimer and Langford, 2009) method. Here, we take a supervised data set $\mathcal{D}^* = \{(x_1, y_1^*) \ldots (x_n, y_n^*)\}$

Name	p(# features)	q(# labels)	n_{train}	n_{test}
Scene	294	6	1211	1196
Yeast	103	14	1500	917
TMC	30438	22	21519	7077
LYRL	47236	4	23149	781265

Table 2: Corpus statistics for different multi-label data sets from the LibSVM repository. LYRL was post-processed so that only top level categories were treated as labels.

Algorithm 1 POEM pseudocode. An alternative version can use separate samplers for estimating ${u_w}^i$ and $\{{u_w}^i\}^2$ on Line 24.

1: **procedure** LossGradient($\mathcal{D}_s$, $\vec{w}$) ▷ Returns ${u_w}^i, \nabla_w({u_w}^i)$ for $i \in \mathcal{D}_s$
2: **for** $i \in \mathcal{D}_s$ **do**
3: $u^i \leftarrow {u_w}^i$. ▷ Equation (13)
4: $g^i \leftarrow \nabla_w({u_w}^i)$.
return $\vec{u}, \vec{g}$.
5: **procedure** ABC($\mathcal{D}$, $\vec{w}$, λ) ▷ Returns A_w, B_w, C_w from Proposition (4)
6: $\vec{u}, \vec{g} \leftarrow LossGradient(\mathcal{D}, \vec{w})$.
7: $R \leftarrow \sum_{i \in \mathcal{D}} u_i / n$.
8: $V \leftarrow \sqrt{\sum_{i \in \mathcal{D}} (u_i - R)^2 / (n-1)}$.
9: $A \leftarrow 1 - \frac{\lambda \sqrt{n} R}{(n-1)V}$.
10: $B \leftarrow \frac{\lambda}{2(n-1)V\sqrt{n}}$.
11: $C \leftarrow \frac{\lambda V}{2\sqrt{n}} + \frac{\lambda \sqrt{n} R^2}{2(n-1)V}$.
return A, B, C.
12: **procedure** SGD($\mathcal{D}$, λ, μ) ▷ $L2$ regularizer μ
13: $\vec{w} \leftarrow [0]_d$. ▷ Initial param
14: $\vec{h} \leftarrow [1]_d$. ▷ Adagrad history
15: **while** True **do**
16: Shuffle $\mathcal{D}$.
17: $A, B, C \leftarrow ABC(\mathcal{D}, w, \lambda)$.
18: **for** $\mathcal{D}_s \subset \mathcal{D}$ **do** ▷ Minibatch $|\mathcal{D}_s| = b$
19: $\vec{u}, \vec{g} \leftarrow LossGradient(\mathcal{D}_s, \vec{w})$.
20: $\overline{u} = \sum_{i \in \mathcal{D}_s} u_i / |\mathcal{D}_s|$.
21: $\overline{g} = \sum_{i \in \mathcal{D}_s} g_i / |\mathcal{D}_s|$.
22: $h_i \leftarrow h_i + {\overline{g}_i}^2$.
23: $j_i \leftarrow \overline{g}_i / \sqrt{h_i}$.
24: $\vec{\nabla} \leftarrow A\vec{j} + 2\mu\vec{w} + 2B\overline{u}\vec{j}$.
25: **if** $\|\vec{\nabla}\| \simeq 0$ **then return** $\vec{w}$. ▷ Gradient norm convergence
26: **if** $\overline{u} >$ avg $\overline{u}$ **then return** $\vec{w}$. ▷ Progressive validation
27: $\vec{w} \leftarrow \vec{w} - \eta\vec{\nabla}$. ▷ Step size η

and simulate a bandit feedback data set from a logging policy h_0 by sampling $y_i \sim h_0(x_i)$ and collecting feedback $\Delta(y_i^*, y_i)$. In principle, we could use any arbitrary stochastic policy as h_0. We choose a CRF trained on 5% of $\mathcal{D}^*$ as h_0 using default hyper-parameters, since they provide probability distributions amenable to sampling. In all the multi-label experiments, $\Delta(y^*, y)$ is the Hamming loss between the supervised label y^* vs. the sampled label y for input x. Hamming loss is just the number of incorrectly assigned labels (both false positives and false negatives). To create bandit feedback $\mathcal{D} = \{(x_i, y_i, \delta_i \equiv \Delta(y_i^*, y_i), p_i \equiv h_0(y_i \mid x_i))\}$, we take four passes through $\mathcal{D}^*$ and sample labels from h_0. Note that each supervised label is worth $\simeq |\mathcal{Y}| = 2^q$ bandit feedback labels. We can explore different learning strategies (e.g., IPS, CRM, etc.) on $\mathcal{D}$ and obtain learnt weight vectors w_{ips}, w_{crm}, etc. On the super-

vised test set, we then report the expected loss per instance $\mathcal{R} = \frac{1}{n_{test}} \sum_i \mathbb{E}_{y \sim h_w(x_i)} \Delta(y_i^*, y)$ and compare the generalization error of these learning strategies.

Baselines and learning methods. The expected Hamming loss of h_0 is the baseline to beat. Lower loss is better. The naïve, variance-agnostic approach to counterfactual learning (Bottou et al., 2013; Strehl et al., 2010) can be generalized to handle parametric multilabel classification by optimizing Equation (12) with $\lambda = 0$. We optimize it either using L-BFGS (IPS($\mathcal{B}$)) or stochastic optimization (IPS($\mathcal{S}$)). POEM($\mathcal{S}$) uses our Iterative-Majorization approach to variance regularization as outlined in Section 5.3, while POEM($\mathcal{B}$) is a L-BFGS variant. Finally, we report results from a supervised CRF as a skyline, despite its unfair advantage of having access to the full-information examples.

We keep aside 25% of $\mathcal{D}$ as a validation set—we use the unbiased counterfactual estimator from Equation (1) for selecting hyper-parameters. $\lambda = c\lambda^*$, where λ^* is the calibration factor from Section 4.4 and $c \in \{10^{-6}, \ldots, 1\}$ in multiples of 10. The clipping constant M is similarly set to the ratio of the $90\%ile$ to the $10\%ile$ propensity score observed in the training set of $\mathcal{D}$. The reported results are not sensitive to this choice of M, any reasonably large clipping constant suffices (e.g. even a simple, problem independent choice of $M = 100$ works well). When optimizing any objective over w, we always begin the optimization from $w = 0$, which is equivalent to $h_w = \text{uniform}(\mathcal{Y})$. We use mini-batch AdaGrad (Duchi et al., 2011) with batch size = 100 and step size $\eta = 1$ to adapt our learning rates for the stochastic approaches and use progressive validation (Blum et al., 1999) and gradient norms to detect convergence. Finally, the entire experiment set-up is run 10 times (i.e. h_0 trained on randomly chosen 5% subsets, $\mathcal{D}$ re-created, and test set performance of different approaches collected) and we report the averaged test set expected error across runs.

6.1 Does variance regularization improve generalization?

Results are reported in Table 3. We statistically test the performance of POEM against IPS (batch variants are paired together, and the stochastic variants are paired together) using a one-tailed paired difference t-test at significance level of 0.05 across 10 runs of the experiment, and find POEM to be significantly better than IPS on each data set and each optimization variant. Furthermore, on all data sets POEM learns a hypothesis that substantially improves over the performance of h_0. This suggests that the CRM principle is practically useful for designing learning algorithms, and that the variance regularizer is indeed beneficial.

6.2 How computationally efficient is POEM?

Table 4 shows the time taken (in CPU seconds) to run each method on each data set, averaged over different validation runs when performing hyper-parameter grid search. Some of the timing results are skewed by outliers, e.g., when under very weak regularization, CRFs tend to take longer to converge. However, it is still clear that the stochastic variants are able to recover good parameter settings in a fraction of the time of batch L-BFGS optimization, and this is even more pronounced when the number of labels grows—the run-time is dominated by computation of $\mathbb{Z}(x_i)$.

$\mathcal{R}$	Scene	Yeast	TMC	LYRL
h_0	1.543	5.547	3.445	1.463
IPS($\mathcal{B}$)	1.193	4.635	2.808	0.921
POEM($\mathcal{B}$)	1.168	4.480	2.197	0.918
IPS($\mathcal{S}$)	1.519	4.614	3.023	1.118
POEM($\mathcal{S}$)	1.143	4.517	2.522	0.996
CRF	0.659	2.822	1.189	0.222

Table 3: Test set Hamming loss, $\mathcal{R}$ for different approaches to multi-label classification on different data sets, averaged over 10 runs. POEM is significantly better than IPS on each data set and each optimization variant (one-tailed paired difference t-test at significance level of 0.05).

Time(s)	Scene	Yeast	TMC	LYRL
IPS($\mathcal{B}$)	2.58	47.61	136.34	21.01
IPS($\mathcal{S}$)	1.65	2.86	49.12	13.66
POEM($\mathcal{B}$)	75.20	94.16	949.95	561.12
POEM($\mathcal{S}$)	4.71	5.02	276.13	120.09
CRF	4.86	3.28	99.18	62.93

Table 4: Average time in seconds for each validation run for different approaches to multi-label classification. CRF is implemented by scikit-learn (Pedregosa et al., 2011). On all data sets, stochastic approaches are much faster than batch gradients.

6.3 Can MAP predictions derived from stochastic policies perform well?

For the policies learnt by POEM as shown in Table 3, Table 5 reports the averaged performance of the deterministic predictor derived from them. For a learnt weight vector w, this simply amounts to applying Equation (11). In practice, this method of generating predictions can be substantially faster than sampling since computing the argmax does not require computation of the partition function $\mathbb{Z}(x)$ which can be expensive in structured output prediction. From Table 5, we see that the loss of the deterministic predictor is typically not far from the loss of the stochastic policy, and often better.

6.4 How does generalization improve with size of $\mathcal{D}$?

As we collect more data under h_0, our generalization error bound indicates that prediction performance should eventually approach that of the optimal hypothesis in the hypothesis space. We can simulate $n \to \infty$ by replaying the training data multiple times, collecting samples $y \sim h_0(x)$. In the limit, we would observe every possible y in the bandit feedback data set, since $h_0(x)$ has non-zero probability of exploring each prediction y. However, the learning rate may be slow, since the exponential model family has very thin tails, and

$\mathcal{R}$	Scene	Yeast	TMC	LYRL
POEM($\mathcal{S}$)	1.143	4.517	2.522	0.996
POEM($\mathcal{S}$)$_{map}$	1.143	4.065	2.299	0.880

Table 5: Mean Hamming loss of MAP predictions from the policies in Table 3. POEM$_{map}$ is significantly better than POEM on all data sets except Scene (one-sided paired difference t-test, significance level 0.05).

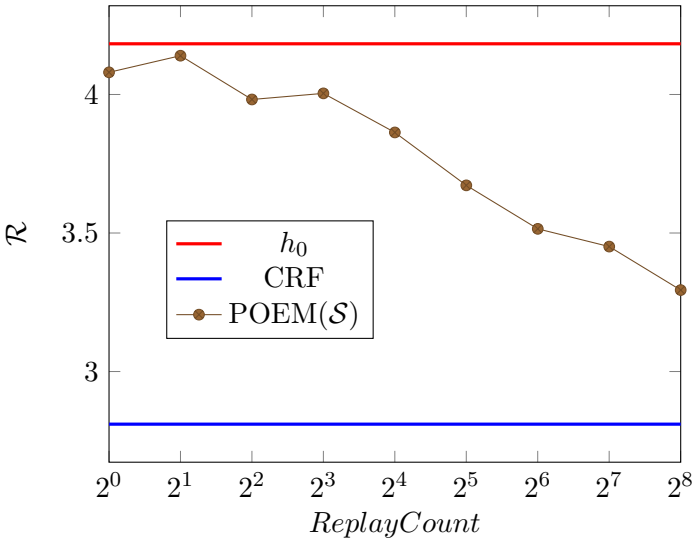

Figure 1: Generalization performance of POEM($\mathcal{S}$) as a function of n on the Yeast data set.

hence may not be an ideal logging distribution to learn from. Holding all other details of the experiment setup fixed, we vary the number of times we replayed the training set (*ReplayCount*) to collect samples from h_0, and report the performance of POEM($\mathcal{S}$) on the Yeast data set in Figure 1. As expected, performance of POEM improves with increasing sample size. Note that even with $\mathit{ReplayCount} = 2^8$, POEM($\mathcal{S}$) is learning from much less information than the CRF, where each supervised label conveys 2^{14} bandit label feedbacks.

6.5 How does quality of h_0 affect learning?

In this experiment, we change the fraction of the training set $f \cdot n_{train}$ that was used to train the logging policy—and as f is increased, the quality of h_0 improves. Intuitively, there's a trade-off: better h_0 probably samples correct predictions more often and so produces a higher quality $\mathcal{D}$ to learn from, but it should also be harder to beat h_0. We vary f from 1% to 100% while keeping all other conditions identical to the original experiment setup in Figure 2, and find that POEM($\mathcal{S}$) is able to consistently find a hypothesis at least as good

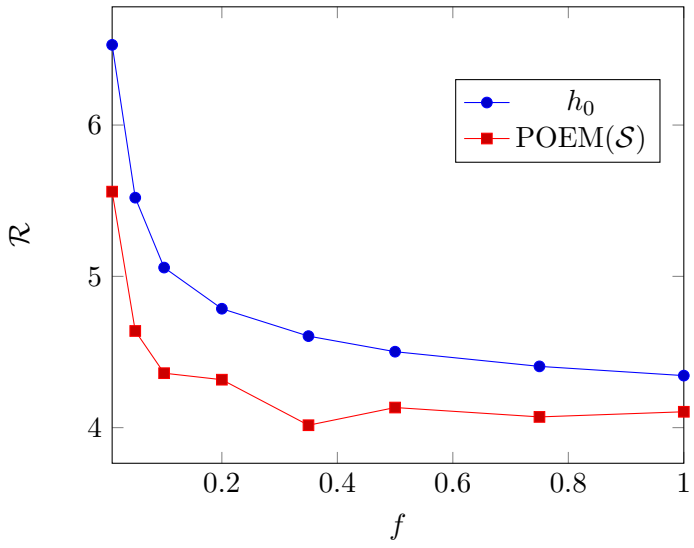

Figure 2: Performance of POEM($\mathcal{S}$) on the Yeast data set as h_0 is improved. The fraction f of the supervised training set used to train h_0 is varied to control h_0's quality. h_0 performance does not reach CRF when $f = 1$ because we do not tune hyper-parameters, and we report its expected loss, not the loss of its MAP prediction.

as h_0. Moreover, even $\mathcal{D}$ collected from a poor quality h_0 ($0.5 \leq f \leq 0.2$) allows POEM($\mathcal{S}$) to effectively learn an improved policy.

6.6 How does stochasticity of h_0 affect learning?

Finally, the theory suggests that counterfactual learning is only possible when h_0 is sufficiently stochastic (the generalization bounds hold with high probability in the samples drawn from h_0). Does CRM degrade gracefully when this assumption is violated? We test this by introducing the *temperature* multiplier $w \mapsto \alpha w, \alpha > 0$ (as discussed in Section 5) into the logging policy. For $h_0 = h_{w_0}$, we scale $w_0 \mapsto \alpha w_0$, to derive a "less stochastic" variant of h_0, and generate $\mathcal{D} \sim h_{\alpha w_0}$. We report the performance of POEM($\mathcal{S}$) on the LYRL data set in Figure 3 as we change $\alpha \in [0.5, \ldots, 32]$, compared against h_0, and the deterministic predictor— h_0 *map*—derived from h_0. So long as there is some minimum amount of stochasticity in h_0, POEM($\mathcal{S}$) is still able to find a w that improves upon h_0 and h_0 *map*. The margin of improvement is typically greater when h_0 is more stochastic. Even when h_0 is barely stochastic ($\alpha \geq 2^4$), performance of POEM($\mathcal{S}$) simply recovers h_0 *map*, suggesting that the CRM principle indeed achieves robust learning.

We observe the same trends (Figures 1, 2 and 3) across all data sets and optimization variants. They also remain unchanged when we include $l2-$regularization (analogous to supervised CRFs to capture the capacity of $\mathcal{H}_{lin}$).

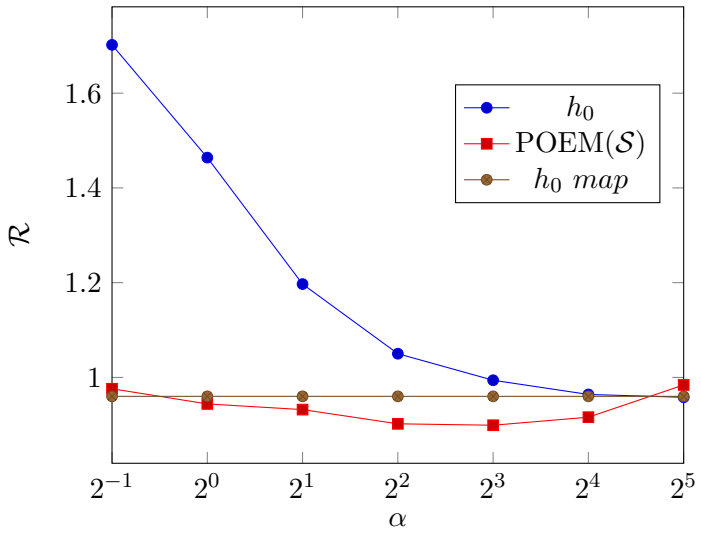

Figure 3: Performance of POEM($\mathcal{S}$) on the LYRL data set as h_0 becomes less stochastic. For $\alpha \geq 2^5$, $h_0 \equiv h_0$ *map* (within machine precision).

7. Real-World Application

We now demonstrate how POEM (and in general the CRM principle) can be instantiated effectively in real world settings. Bloomberg, the financial and media company in New York, had the following challenging retrieval problem: the task was to train a high-precision classifier that could reliably pick the best answer d^* (or none, if none answered the query) from a pool of candidate answers $\mathcal{Y}(x)$ for query x, where $\mathcal{Y}(x)$ was generated by an existing high-recall retrieval function. The challenge lay in collecting supervised labeled data that could be used to train this high-precision classifier.

Before we started our experiment with POEM, an existing high-precision classifier was already in operation. It was trained using a few labeled examples (x, d^*), but scaling up the system to achieve improved accuracy appeared challenging given the cost of acquiring new (x, d^*) pairs that mimicked what the system saw during its operation. However, it was possible to collect logs of the system, where each entry contained a query x and the features $\phi(x, d)$ describing each candidate answer $d \in \mathcal{Y}(x)$. The high-precision classifier could be modeled as a logistic regression classifier with weights w and a threshold τ. Each candidate was scored using w, $s(d) = w \cdot \phi(x, d)$. If the highest scoring candidate $s(d^*) \geq \tau$, it was selected as the answer and otherwise the system abstained.

This existing system could easily be adapted to provide $\mathcal{D}$ as needed by POEM. For each x, a dummy $d_0 \in \mathcal{Y}(x)$ is added to the candidate pool to model abstention. During the operation of the system, answers are *sampled* according to $\frac{\exp(\alpha \cdot s(d))}{\mathbb{Z}}$. $\mathbb{Z}$ is the partition function to ensure this is a valid sampling distribution, $\mathbb{Z} = \sum_{d \in \mathcal{Y}(x) \cup d_0} \exp(\alpha \cdot s(d))$. Abstention is modeled by the fact that d_0 is sampled with probability proportional to $\exp(\alpha \cdot s(d_0))$. α is a temperature constant so that the system can be tuned to sample abstentions at

roughly the same rate as its deterministic counterpart. Finally, the end-result feedback ($\delta \in \{\texttt{thumbs-up}, \texttt{thumbs-down}\}$ represented as binary feedback) was logged and provided bandit feedback for the presented answer d.

This data set was much easier to collect during the system run compared to annotating each x in the logs with the best possible d^* that would have answered the query. We argue that this is a general, practical, alternative approach to training retrieval systems: use any strategy with very high recall to construct $\mathcal{Y}$, then use the parameters w estimated using the CRM principle to search through this $\mathcal{Y}$ and find a precise answer.

On a small pilot study, we acquired $\mathcal{D}$ with $\simeq 4000$ $(x, d, \frac{\exp(\alpha \cdot s(d))}{\mathbb{Z}}, \delta)$ tuples in the training set and $\simeq 500$ tuples in the validation and test sets. We verified that the existing high-precision classifier was statistically significantly better than random baselines for the problem. POEM($\mathcal{S}$) is trained on this log data by performing gradient descent with w initialized to $w_0 = 0$ and validating $c \in \left[10^{-6}, \ldots 1\right], \lambda = c\lambda^*$ as described in Sections 4.4 and 6. POEM($\mathcal{S}$) found a w^* that improved δ feedback over the existing system by over 30%, as estimated using the unbiased counterfactual estimator of Equation (1) on the test set. Without using the variance regularizer, the IPS($\mathcal{S}$) found a w^* that degraded the system performance by 3.5% estimated counterfactually in the same way. This shows that POEM and the CRM principle can bring potential benefit even in binary-feedback multi-class classification settings where classic supervised learning approaches lack available data.

8. Conclusion

Counterfactual risk minimization serves as a robust principle for designing algorithms that can learn from a batch of bandit feedback interactions. The key insight for CRM is to expand the classical notion of a hypothesis class to include stochastic policies, reason about variance in the risk estimator, and derive a generalization error bound over this hypothesis space. The practical take-away is a simple, data-dependent regularizer that guarantees robust learning. Following the CRM principle, we developed the POEM learning algorithm for structured output prediction. POEM can optimize over rich policy families (exponential models corresponding to linear rules in supervised learning), and deal with massive output spaces as efficiently as classical supervised methods.

The CRM principle more generally applies to supervised learning with non-differentiable losses, since the objective does not require the gradient of the loss function. We also foresee extensions of the algorithm to handle ordinal or co-active feedback models for $\delta(\cdot, \cdot)$, and extensions of the generalization error bound to include adaptive or deterministic h_0, etc.

Acknowledgments

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

References

Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, New York, NY, USA, 2009.

Alina Beygelzimer and John Langford. The offset tree for learning with partial labels. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 129–138, 2009.

Avrim Blum, Adam Kalai, and John Langford. Beating the hold-out: Bounds for k-fold and progressive cross-validation. In *Proceedings of the Twelfth Annual Conference on Computational Learning Theory*, pages 203–208, 1999.

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.

Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, pages 442–450, 2010.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. In *Proceedings of the 28th International Conference on Machine Learning*, pages 1097–1104, 2011.

Nicolas Galichet, Michèle Sebag, and Olivier Teytaud. Exploration vs exploitation vs safety: Risk-aware multi-armed bandits. In *Asian Conference on Machine Learning*, pages 245–260, 2013.

J. Garcia and F. Fernandez. Safe exploration of state and action spaces in reinforcement learning. *Journal of Artificial Intelligence Research*, 45:515–564, 2012.

Katja Hofmann, Anne Schuth, Shimon Whiteson, and Maarten de Rijke. Reusing historical interaction data for faster online learning to rank for IR. In *Sixth ACM International Conference on Web Search and Data Mining*, pages 183–192, 2013.

Edward L. Ionides. Truncated importance sampling. *Journal of Computational and Graphical Statistics*, 17(2):295–311, 2008.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289, 2001.

John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Proceedings of the 21st Annual Conference on Neural Information Processing Systems*, pages 817–824, 2008.

John Langford, Alexander Strehl, and Jennifer Wortman. Exploration scavenging. In *Proceedings of the 25th International Conference on Machine Learning*, pages 528–535, 2008.

Adrian S. Lewis and Michael L. Overton. Nonsmooth optimization via quasi-newton methods. *Mathematical Programming*, 141(1-2):135–163, 2013.

Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, pages 661–670, 2010.

Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, pages 297–306, 2011.

Lihong Li, Shunbao Chen, Jim Kleban, and Ankur Gupta. Counterfactual estimation and optimization of click metrics for search engines. *CoRR*, abs/1403.1891, 2014.

Lihong Li, Remi Munos, and Csaba Szepesvari. Toward minimax off-policy value estimation. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

Jérémie Mary, Philippe Preux, and Olivier Nicol. Improving offline evaluation of contextual bandit algorithms via bootstrapping techniques. In *Proceedings of the 31st International Conference on Machine Learning*, pages 172–180, 2014.

Andreas Maurer and Massimiliano Pontil. Empirical bernstein bounds and sample-variance penalization. In *Proceedings of the 22nd Conference on Learning Theory*, 2009.

Art B. Owen. *Monte Carlo theory, methods and examples.* 2013.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Paul R. Rosenbaum and Donald B. Rubin. The central role of the propensity score in observational studies for causal effects. *Biometrika*, 70(1):41–55, 1983.

Pannagadatta K. Shivaswamy and Thorsten Joachims. Multi-armed bandit problems with history. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics*, pages 1046–1054, 2012.

Alexander L. Strehl, John Langford, Lihong Li, and Sham Kakade. Learning from logged implicit exploration data. In *Proceedings of the 24th Annual Conference on Neural Information Processing Systems*, pages 2217–2225, 2010.

Adith Swaminathan and Thorsten Joachims. Counterfactual risk minimization: Learning from logged bandit feedback. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

Philip S. Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High-confidence off-policy evaluation. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 3000–3006, 2015.

Ioannis Tsochantaridis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21st International Conference on Machine Learning*, pages 104–, 2004.

G.J.J. van den Burg and P.J.F. Groenen. GenSVM: A Generalized Multiclass Support Vector Machine. Technical Report EI 2014-33, Erasmus University Rotterdam, Erasmus School of Economics (ESE), Econometric Institute, 2014.

W. N. Wapnik and A. J. Tscherwonenkis. *Theorie der Zeichenerkennung*. Akademie Verlag, Berlin, 1979.

Jin Yu, S. V. N. Vishwanathan, Simon Günter, and Nicol N. Schraudolph. A quasi-Newton approach to nonsmooth convex optimization problems in machine learning. *Journal of Machine Learning Research*, 11:1145–1200, 2010.