A Hybrid Framework of Counterfactual Risk Minimization and Supervised Learning on Logged Bandit Feedback

Jinning Li
Department of Computer Science
Shanghai Jiao Tong University*
lijinning@sjtu.edu.cn

Thorsten Joachims
Department of Computer and Information Science
Cornell University
tj@cs.cornell.edu

Abstract

Learning from the logged bandit feedback is an important task in machine learning. Some previous methods minimize the counterfactual risk with batch learning algorithm and the others reduce this task into a prediction problem of supervised learning. However, both of these two kinds of methods have their own advantages and disadvantages. In this paper, we will introduce a hybrid framework combining the advantages of these two kinds of methods. In this new framework, the difference between the observed feedbacks and our predictions are minimized. The importance sampling of propensities of logging policy and new policy is used as the weighting for the loss function of supervised learning. Experimental result proves that this new framework can improve the policy learning performance on the Criteo dataset by reducing bias and variance.

Introduction

The logged data can be find almost everywhere in our daily life. For example, the interaction between the user and an item recommendation system will be recorded. These logged data can be viewed as knowledge basis. How to learn from these knowledge basis is an interesting and important task in machine learning. In this paper, we will focus on a kind of data format named logged bandit feedback recorded from interactive systems.

Many previous works are already capable of learning from the logged bandit feedback. There are mainly two kinds of method. The first kind is to directly learn a new policy system minimize a measure named counterfactual risk, such as the banditnet proposed in (Joachims, Swaminathan, and de Rijke 2018). The second kind is to reduce this policy learning task into supervised learning. A network is trained to predict the unseen feedbacks with the observed feedbacks. Then a new policy is built based on the predicted feedbacks, such as the model introduced in (Beygelzimer and Langford 2009).

However, both of these methods have their advantages and disadvantages. For example, the counterfactual risk minimization methods will face a problem of bias-variance tradeoff. It is also hard to bound its variance and to select a suitable hyper-parameter. The supervised learning method can only estimates the policy indirectly. So the performance of this kind of method will be limited if the logging policy is complex or only limited candidates are preferred and displayed by logging policy.

In this paper, we introduce a hybrid framework of counterfactual risk minimization and supervised learning. The difference between the observed feedbacks and our predictions are minimized. The importance sampling of propensities of logging policy and new policy is also used as the weighting for the loss function of supervised learning. Experiment proves that this hybrid framework can reduce the bias while control the variance. Clipping and variance can also be applied to the objective of hybrid learning for better performance.

Preliminaries

Logged Bandit Feedback

The logged data of bandit feedbacks exists everywhere. It can be recorded by the systems such as search engines, ad placements and recommendation systems. Assume that the input of a system is the features of some samples, denoted by $x \in \mathcal{X}$. The output is a prediction $y \in \mathcal{Y}$. For example, in the item recommendation system, the input x can be the features of items and users, and the prediction y can be the ranking from user. We assume the input x is drawn from an fixed but underlying distribution $P(\mathcal{X})$. An hypothesis space $\mathcal{H}$ is assigned to the learner. A hypothesis $h(\mathcal{Y}|x) \in \mathcal{H}$ is a mapping from the input space to the output space $\mathcal{Y}$. The hypothesis will make predictions by sampling $y \sim h(\mathcal{Y}|x)$. For the convenience, we use $h(x)$ to denote $h(\mathcal{Y}|x)$.

However, in the setting of interactive learning system, we can only observe the feedback $\delta(x, y)$ for the y sampled from $h(x)$. For example, in the item recommendation system, we can only observe the ranking of user after we recommend an item for the user. The objective of an item recommendation system is to find a hypothesis $h \in \mathcal{H}$ maximizing the ranking or minimize the risk, or say, expected loss:

$$R(h) = \mathbb{E}_{x\sim P(\mathcal{X})}\mathbb{E}_{y\sim h(x)}[\delta(x, y)] \quad (1)$$

During the interaction between user and recommendation system, we assume the hypothesis h is unchanged. We use π_0 to denote the logging policy using this fixed hypothesis. The input features x, the prediction, or say, action

*This project is advised by Professor Thorsten Joachims when the author visits Cornell University as a research intern.

$y \sim \pi_0(\mathcal{Y}|x)$, and the feedback δ can be logged for training a new policy training. We also assume that the propensity for the system to make prediction y is also recorded as $p = \pi_0(y|x)$. The data collected from the system is:

$$\mathcal{D} = \{(x_1, y_1, \delta_1, p_1), ..., (x_n, y_n, \delta_n, p_n)\} \tag{2}$$

Counterfactual Risk Minimization

The objective of learning of logged bandit feedback is to learn a new policy π_w that minimize the distribution mismatch between π_0 and π_w. That is to minimize an unbiased estimate of $R(\pi_w)$:

$$R(\pi_w) = \frac{1}{n} \sum_{i=1}^{n} \delta_i \frac{\pi_w(y_i|x_i)}{p_i} \tag{3}$$

However, it is not practical to optimize this equation directly. There are three reasons. First, this strategy is not invariant to additive transformation of the logged loss. The gradient will be significantly affected if the loss is not scaled properly. Second, this estimator has unbounded variance. A constraint to limit the variance of the new policy is required. Third, this equation is not capable to estimate $R(\pi_w)$ for different policies because they may have vastly different variance.

There are mainly two kinds of machine learning methods to solve this task. The first kind is applying batch learning method to minimizing this counterfactual risk. The second kind is reduce this counterfactual risk minimization task into a supervised learning task. We will introduce an batch learning method named Policy Optimizer for Exponential Models (POEM) algorithm in Section . We will introduce how to reduce this task into supervised learning and some variants of supervised learning in Section . In Section , we will introduce how to hybrid the counterfactual risk minimizing methods with supervised learning methods in Section .

Batch Learning: Policy Optimizer for Exponential Models

The idea of the Policy Optimizer for Exponential Models (POEM) (Swaminathan and Joachims 2015) is to minimizing the counterfactual risk estimation directly while applying different constraints such as regularizer to control the bias and variance. POEM is proved to be effective by solving the three problems mentioned above. The objective of POEM is

$$\arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \delta_i \min\{M, \frac{\pi_w(y_i|x_i)}{\pi_0}\} + \lambda \sqrt{\frac{Var_w(x)}{n}}, \tag{4}$$

where w is the parameters of the learner such as neural networks. n is the number of samples in training dataset. δ_i is the logged feedback by logging policy π_0. π_w is the new policy we are training. $\sqrt{\frac{Var_w(x)}{n}}$ is the standard deviation regularizer scaled by the hyper-parameter λ. The minimization operation $\min M, \cdot$ here is the clipping operation with a threshold M.

In Eqn.4, $\pi_w(y_i|x_i)$ is the propensity of pi_w for the action y_i under the condition of input contextual features x_i. $\pi_w(y_i|x_i)$ can be estimated by a Softmax function

$$\pi_w(y_i|x_i) = \frac{\exp(w \cdot \phi(x_i, y_i))}{\sum_{y_i' \in \mathcal{Y}_i} \exp(w \cdot \phi(x_i, y_i'))}, \tag{5}$$

where $\phi(\cdot)$ is a combination function, for example, the concatenation operation. $\phi(x_i, y_i)$ is the joins of contextual features of input x_i and action y_i.

The clipping operation is a bias-variance tradeoff proceess. It will reduce the variance of our model but slightly increase the bias. Well balanced bias and variance are important for the improvement of our new policy π_w. The standard deviation regularizer mentioned above is also capable of bounding the variance of π_w. All these constraints helps the learning of new policy to minimize the distribution mismatch between π_w and π_0.

Supervised Learning: Variants

Another effective method to solve the counterfactual risk minimization task is to reduce it into an supervised learning task. In the setting of logged bandit feedback, we can only observe the feedback of the actions which we already chose to display. The idea of using supervised learning is to train a learner such as neural networks or linear regression with already observed feedbacks, to predict the unseen feedbacks. Then, we can manually select a policy based on these predicted feedbacks. In another word, the learner is trained to select a hypothesis $h \in \mathcal{H}$ to fit the observed feedback δ_i

$$h = \arg\min_{h_w} \sum_{i=1}^{n} \mathcal{L}(\delta_i, h_w(x_i, y_i)), \tag{6}$$

where $\mathcal{L}$ is the expected loss between the logged feedback δ_i and our predictions. With the feedback predictions, we can manually select a new policy such as choosing the actions with largest feedback

$$\hat{\pi}(y_i|x_i) = \arg\max_{y_i \in \mathcal{Y}} (h(x_i, y_i)) \tag{7}$$

The most straightforward method is to minimize the cross entropy loss between the logged bandit feedback and our predictions

$$\mathcal{L}_h = -\left[\delta \cdot \log h(x, y) + (1 - \delta) \cdot \log(1 - h(x, y))\right]. \tag{8}$$

Inverse Propensity Weighting

Reducing the counterfactual risk minimization problem into supervised learning directly will cause bias. Because we only observe the feedback of candidate selected by π_0. Since the logging policy π_0 will always choose some actions having high propensities. Therefore these actions are more likely to appear in the logged data. The sampling of logged data will become unbalanced. This problem will affects the bias of our new policy significantly. In order to solve this problem, we introduce the inverse propensity method to our objective.

$$\mathcal{L}_h = -\frac{1}{p_0}\left[\delta \cdot \log h(x, y) + (1 - \delta) \cdot \log(1 - h(x, y))\right]. \tag{9}$$

Although the propensity weighting can reduce the bias caused by unbalanced sampling, it will leads to another two problems. First, propensity weighting will cause high variance because the logged propensity p_0 can be extremely small. The variance of the inverse propensity is also unbounded. Second, we know that in the logged bandit feedback, the propensity $p_0(y_i|x_i)$ of an action y_i is estimated by π_0 within an action space $\mathcal{Y}_\rangle$. the propensity of action y_i is different with respect to different action space $\mathcal{Y}_\rangle$. It's not a global propensity for the action y_i. It's not suitable to weight the expected loss with inverse propensity directly. Note that this problem does not affect the POEM model because POEM is trains in batches of every action space $\mathcal{Y}_\rangle$.

Inverse Propensity Weighting with Clipping

The most effective way to solve the problem of unbounded variance is applying clipping to the inverse propensity weighting. The expected loss can be expressed as

$$\mathcal{L}_h = -\min\{M, \frac{1}{p_0}\}[\delta \cdot \log h(x,y) \tag{10}$$

$$+ (1-\delta) \cdot \log(1-h(x,y))], \tag{11}$$

where the hyper-parameter M is the clipping threshold.

The clipping method is effective because it will control the gradient of back-propagation when $\frac{1}{p_0}$ goes too large. Practically, $\frac{1}{p_0}$ can often goes extremely large because if all the action $y_i \in \mathcal{Y}_\rangle$ are negative samples, the propensity can be unlimited closed to 0.

Hybrid method: Counterfactual Supervised Learning

In order to solve the second problem mentioned in Section , we use the important sampling to weight the loss function. That is, we introduce the propensity of new policy into the numerator of inverse propensity. The new loss function is

$$\mathcal{L}_h = -\frac{p_w}{p_0}\left[\delta \cdot \log h(x,y) + (1-\delta) \cdot \log(1-h(x,y))\right]. \tag{12}$$

This kind of weighting solves the problem that p_0 is bounding with specific action space $\mathcal{Y}_\rangle$, because the value $\frac{1}{p_0}$ is balanced with the new propensity p_w also estimated within the action space $\mathcal{Y}_\rangle$. We can use the Softmax function to estimate p_w by

$$p_w(y_i|x_i) = \frac{exp[h(x_i,\hat{y}_i) - \max h(x_i,y_i)]}{\sum_{y_i \in \mathcal{Y}_i} exp[h(x_i,y_i) - \max h(x_i,y_i)]} \tag{13}$$

This method can also reduce the variance since the value of $\frac{1}{p_0}$ can be bounded by p_w to a great extent.

Compared with the objective of POEM in Eqn.4 and the straightforward supervised learning in Eqn.8, this hybrid model not only minimizes the feedback difference of the action y_0 selected by p_0, but also applies batch learning for all the actions in action space $\mathcal{Y}_i$. This process is shown in Figure 1.

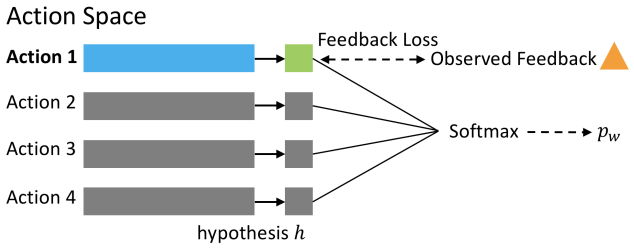

Figure 1: Loss calculating of hybrid learning method. This action space includes 4 candidate actions. Action 1 is selected for display by π_0. First, the hybrid learning minimizes the difference between the predicted feedback and observed feedback of action 1. Second, the hybrid learning calculate the relative propensity p_w by applying Softmax function to all candidate actions.

Applying clipping and variance regularizer

The clipping operation and variance regularizer can also be applied to the objective of hybrid learning method. The general representation of loss function can be written as

$$\mathcal{L}_h = -\min\{M, \frac{p_w}{p_0}\}[\delta \cdot \log h(x,y) \tag{14}$$

$$+ (1-\delta) \cdot \log(1-h(x,y))] + \lambda Reg. \tag{15}$$

For example, the regularizer Reg can be tanh-MSE difference between $\frac{1}{p_w}$ and $\frac{1}{p_0}$ and the regularizer used by POEM $\sqrt{\frac{Var_w(x)}{n}}$.

Experiments

We operate the Experiment based on the dataset from NIPS'17 Workshop: Criteo Ad Placement Challenge (Lefortier et al. 2016) on CrowAI platform. All the codes are avaliable here: `https://github.com/jinningli/ad-placement-pytorch`.

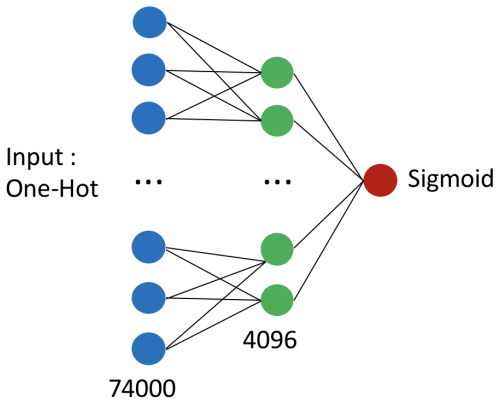

Figure 2: Multiple layer perceptron used in our experiments. The first layer contains 74000 dimensions and the second layer contains 4096 dimensions.

In our experiments, an two layer perceptron is adopted as the hypothesis space $\mathcal{H}$. The feature is encoded as One-

Hot in order to be inputted to the networks. Note that Sigmoid function is not applied in the setting of hybrid learning method.

Criteo Ad Placement Dataset

The dataset of the Criteo Ad Placement dataset used in our experiments follows the data format mentioned in Eqn. 2. In the setting of Criteo dataset, the logging policy π_0 represents the Ad placement policy used by Criteo company. x_i is the context of Ad candidates. y_i represents which candidate is chosen to display, p_i is the corresponding propensity. The feedback is whether the Ad is clicked by user. If it's clicked, δ_i is equal to one. The action space $\mathcal{Y}_i$ is called impression. An impression is an Ad position with several Ad candidates.

Metrics

There are mainly two evaluation metrics of the Criteo dataset. The first one is the inverse propensity score (IPS), represented by

$$IPS = \frac{10^4}{n^+ + 10n^-} \sum_{i=1}^{n} \delta_i \frac{\pi_w(\hat{y_i}|x_i)}{\pi_0} \tag{16}$$

where $\pi_w(\hat{y_i}|x_i)$ is the new policy developed by our algorithm. $\pi_w(\hat{y_i}|x_i)$ is calculated by

$$\pi_w(\hat{y_i}|x_i) = \frac{exp[h(x_i, \hat{y_i}) - \max h(x_i, y_i)]}{\sum_{y_i \in \mathcal{Y}_i} exp[h(x_i, y_i) - \max h(x_i, y_i)]}, \tag{17}$$

In this formula, the $n^+ + 10n^-$ is to balance the number of positive and negative samples. We can ignore the factor 10^4

The second metric is the standard deviation of IPS, the formulation is

$$\frac{2.58 \times \sqrt{n}}{n^+ + 10n^-} \times Std[\delta \frac{\pi_w(\hat{y_i}|x_i)}{\pi_0}] \tag{18}$$

It's the standard deviation of IPS on all the impressions.

Baseline on Criteo Dataset: Follow the Regularized Leader

Follow the Regularized Leader (FTRL) algorithm (McMahan 2011) is first proposed by Google in 2003. It's used by the Rank 1 of the Criteo Ad Placement Challenge in 2007. FTRL is an online learning method which is adapted for faster and better processing for sparse feature representations. However, we're not going to introduce the detail of FTRL. What really interests us is his post-processing method

$$h(x, y) = \frac{850100}{1 + e^{-h(x,y)+1.1875}}$$

In this paper, we will introduce why his post-processing is so critical. This explains why other participants got much lower scores.

The reason is Most participants treat this challenge as a CTR prediction challenge. So, their output $\tilde{\delta} \in [0, 1]$. However, the Softmax function in Eqn.17 is applied to get π_w, which is embedded in the evaluation program.

Using post-processing, their π_w in evaluation program look like: [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 3.05902e-07]. This π_w is more reasonable because it has a clear preference for the candidate with propensity 1.

However, without post-processing, their π_w in evaluation program look like [0.984536, 0.980575, 0.984368, 0.974204, 0.97577, 0.986368, 0.976827, 1, 0.984053, 0.995168, 1]. The propensity for all of the candidates are almost the same. So this π_w is almost the same to random selection.

Effectiveness of Clipping

we investigated how the clipping method affects the bias-variance tradeoff for the inverse propensity weighting with clipping model introduced in Section .

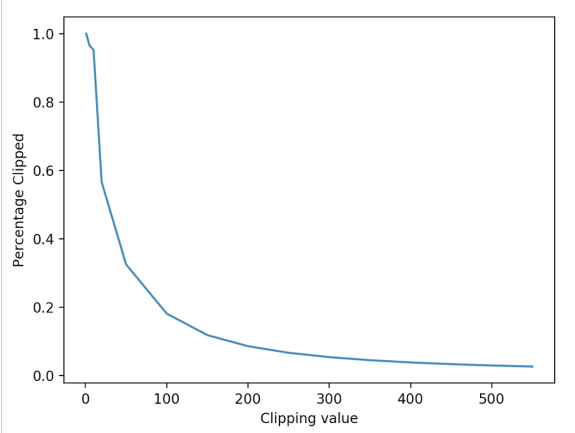

Figure 3: The percentage of clipped inverse propensity changes with the clipping threshold M.

We first analyze the distribution of inverse propensity on the Criteo dataset. Figure 3. When $M = 1$, all the inverse propensities are clipped because inverse propensity is always larger than 1. The problem degenerates to straightfoward supervised learning in Section When $M = +\infty$, no inverse propensities are clipped. The problem degenerates to non-clipping model in Section . According to the distribution shown in Figure 3, we selected M=[1, 2, 5, 10, 15, 20, 30, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500, 550] to see how the IPS score varies.

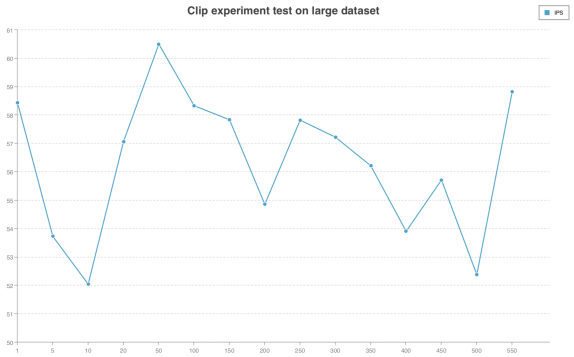

Figure 4: The inverse propensity score under different M.

The IPS value under different threshold M is shown in Figure 4. We can find that without propensity weighing, or say, threshold is 1, the IPS is 58.4. When $M = +\infty$, no inverse propensities are clipped. The IPS value is about 54.55.

The top IPS appears when $M = 30$. The IPS is about 60.6, which is better than two extreme M. So The clipping can improve the performance of our model.

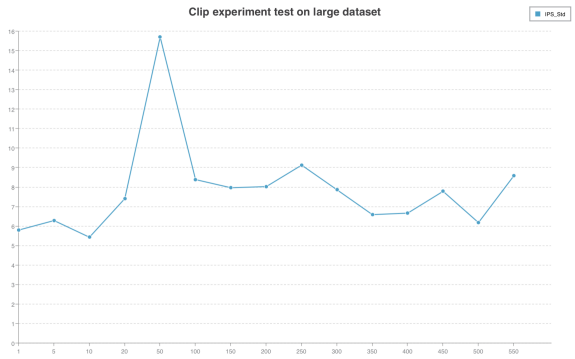

Figure 5: The standard deviation of inverse propensity score under different M.

The standard deviation is shown in Figure 5. An interesting thing is that, it seems the higher IPS is, the higher standard deviation it receives. Maybe this is because the predict for the negative samples becomes more accurate, so there will be more IPS whose value is zero. The standard deviation is thus increased.

Inverse Propensity Scoring and Standard Deviation

We test the inverse propensity scores and their corresponding standard deviations for different models mentioned in this paper. All the result is shown in Table.1.

Table 1: IPS and Standard Deviation of IPS on different models

Model	IPS	IPS Std
FTRL	55.70	4.30
Straightforward Supervised Learning	59.25	5.46
Inverse Propensity Weighting	54.55	2.94
Inverse Propensity with Clipping	60.60	15.80
Hybrid Learning	**61.69**	27.06
Hybrid (no gradient for π_w)	52.95	**2.42**
Hybrid with tanh-MSE regularizer	52.46	5.41

The highest IPS I got is from Hybrid Learning model and the smallest standard deviation comes from Hybrid Learning without no gradient of π_w. The result shows that the Straightforward Supervised Learning model is already capable of learning a good Ad placement policy with suitable hyper-parameters. Pure Inverse Propensity Weighting model does not perform well. However, clipping helps improves the performance of Pure Inverse Propensity Weighting model. After applying tanh-MSE regularizer, the IPS of Hybrid Learning is reduced. This can be attributed to improper regularizer or value of λ.

Conclusion

Logged bandit feedback is an important data format recorded on the interactive systems such as item recommendation, Ad placement, and search engines. In this paper, we propose a hybrid learning method combining counterfactual risk minimization and supervised learning on logged bandit feedback. We also investigate the effectiveness of post-processing and clipping method. Experimental result prove that hybrid learning can help reduce both of the bias and variance. Higher inverse propensity scoring and lower standard deviation is obtained by hybrid learning method. In the future work, experiments of more previous works should be operated to compare with our model. We should also try to use another dataset for experiments. We can also try other network architectures such as convolutional neural networks.

Acknowledgement

This work was supported in part by NSF-1513692 and NSF-1615706. Thanks Professor Thorsten Joachims for his careful advising and innovative ideas. Thanks Nayun Xu for discussing and cooperating with me on this project.

References

Beygelzimer, A., and Langford, J. 2009. The offset tree for learning with partial labels. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 129–138. ACM.

Joachims, T.; Swaminathan, A.; and de Rijke, M. 2018. Deep learning with logged bandit feedback.

Lefortier, D.; Swaminathan, A.; Gu, X.; Joachims, T.; and de Rijke, M. 2016. Large-scale validation of counterfactual learning methods: A test-bed. *arXiv preprint arXiv:1612.00367*.

McMahan, B. 2011. Follow-the-regularized-leader and mirror descent: Equivalence theorems and l1 regularization. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 525–533.

Swaminathan, A., and Joachims, T. 2015. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*, 814–823.