Discriminative Adversarial Learning: A General Framework for Interactive Learning

Yue Wang

1 Introduction

Recent developments in generative adversarial networks (GAN) show that state-of-the-art generative models can be trained in a framework of minimax game [7, 18]. In this game, a generator tries to generate synthetic data examples that are very similar to organic ones, and a discriminator (a binary classifier) tries to distinguish which examples are organic and which are synthetic. In the theoretical equilibrium where both parties have sufficiently large model capacity and computational power, the generator generates data from the true data distribution, while the discriminator can only perform random guess.

Inspired by this framework, we formulate discriminative model (classifier) training in a minimax game. In this game, a generator tries to generate training examples that are hard to classify, while the discriminator (the classifier) tries to correctly classify examples provided by the generator. In the theoretical equilibrium where both parties have sufficiently large capacity and computational power, the generator identifies the Bayes decision boundary and only draw training examples from there, while the discriminator can only perform random guess.

Both research and practice of applied machine learning show increasing demand for learning algorithms that can interact with human users. A game-theoretic setup naturally depicts learning as an interactive, turn-based, and continuous process, which can facilitate algorithmic design better than the conventional non-interactive, one-off setup of supervised learning. The goal of this work is three-fold:

(1) To formulate and analyze a new game as a theoretical foundation for discriminative model learning algorithms, which is connected to various existing algorithms including SVMs and boosting.

(2) To propose a general active learning framework entailed from the game-theoretic setup, which unifies various existing active learning strategies such as uncertainty sampling and density-based sampling.

(3) To draw implications for interactive machine learning algorithm design, such as ReQ-ReC [12], feature labeling [3], and pseudo-task learning.

2 A minimax game

For clarity of presentation, let us consider the simplest setting of discriminative learning: binary classification. Suppose our data (x, y) comes from an underlying joint distribution P_{XY}, $x \in \mathcal{X} \subset \mathbb{R}^d$ and $y \in \mathcal{Y} = \{0, 1\}$. When conditioning on x, we obtain a posterior distribution $\eta(x) = \Pr(Y = 1|X = x)$. For any given classifier $h : \mathcal{X} \to \mathcal{Y}$, we define the loss function as

$$R(P_{XY}, h) = \mathbb{E}_{(x,y)\sim P_{XY}} \left[\mathbf{1}_{\{h(x)\neq y\}}\right] \tag{1}$$

which uses zero-one loss. Note that this loss function depends on both the data distribution P_{XY} and the classifier h. When h has sufficient capacity, $R(P_{XY}, h)$ is minimized by the Bayes classifier

$$h^*(x) = \begin{cases} 1 & \text{if } \eta(x) \geq 1 - \eta(x); \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The two players in a **discriminative adversarial game** are generator G and discriminator D.

- The generator G selects a probability distribution $g \in \mathcal{G}$, where $\mathcal{G}$ is a class of probability distributions over input data space $\mathcal{X}$. G draws unlabeled data from g, and for each unlabeled data x it draws a binary label from $\eta(x)$. By choosing g, G defines a new joint distribution G_{XY} over $\mathcal{X} \times \mathcal{Y}$. If g is uniform, $G_{XY} = P_{XY}$. The goal of G is to draw examples on which D has a high classification error.

- The discriminator D selects a classifier $h \in \mathcal{H}$, where $\mathcal{H}$ is a class of functions mapping from $\mathcal{X}$ to $\mathcal{Y}$. The goal of D is to learn to correctly classify the examples given by G.

G and D play the following two-player minimax game with value function $R(G_{XY}, h)$:

$$\min_D \max_G R(G_{XY}, h) = \mathbb{E}_{(x,y)\sim G_{XY}} \left[\mathbf{1}_{\{h(x)\neq y\}}\right] \ . \tag{3}$$

2.1 Theoretical analysis

In the theoretical analysis, we assume that g and h can take arbitrary distribution/function form (their model classes have sufficient capacity).

Theorem 1. *For fixed distribution selected by G, the optimal discriminator D is the Bayes classifier h^*.*

Proof. For any h,

$$\begin{aligned} R(G_{XY}, h) &= \mathbb{E}_{(x,y)\sim G_{XY}} \left[\mathbf{1}_{\{h(x)\neq y\}}\right] & (4)\\ &= \mathbb{E}_{x\sim g(x)} \left[\mathbb{E}_{y\sim \eta(x)} \left[\mathbf{1}_{\{h(x)\neq y\}}\right]\right] & (5)\\ &= \mathbb{E}_{x\sim g(x)} \left[\eta(x)\mathbf{1}_{\{h(x)=0\}} + (1-\eta(x))\mathbf{1}_{\{h(x)=1\}}\right] & (6) \end{aligned}$$

To minimize $R(G_{XY}, h)$ for fixed G_{XY}, it suffices for $h(x)$ to be such that $\forall x$,

$$\eta(x)\mathbf{1}_{\{h(x)=0\}} + (1-\eta(x))\mathbf{1}_{\{h(x)=1\}} \tag{7}$$

is minimized. Note that the indicators here are mutually exclusive, so it suffices for D to choose the Bayes classifier $h^*(x)$ defined in Eq. (2). The discriminator does not need to be defined outside of $\mathrm{supp}(g)$[1], concluding the proof. □

The above theorem holds because Bayes classifier h^* is agnostic to X-marginal distribution. The minimax game in Eq. (3) can then be reformulated as:

$$\begin{aligned} \max_G L(G) &= \max_G R(G_{XY}, h^*) & (8)\\ &= \max_g \mathbb{E}_{x\sim g(x)} \left[r(x)\right] & (9) \end{aligned}$$

where we define the risk function $r(x) = \min(\eta(x), 1-\eta(x))$.

Theorem 2. *The global maximum of generator G's objective* (9) *is achieved if and only if the $g(x)$ has non-zero probability density where $\eta(x) = 1/2$. That is, $\forall x \in \mathrm{supp}(g)$, $\eta(x) = 1/2$. The value of the minimax game is* $1/2$.

Proof. Note that $r(x)$ reaches its maximum when $\eta(x) = 1-\eta(x)$, i.e. $\eta(x) = 1/2$.

$$\max_G L(G) = \max_g \mathbb{E}_{x\sim g(x)} \left[r(x)\right] \leq \max_x r(x) = \frac{1}{2} \ . \tag{10}$$

The equality holds if $g(x)$ has non-zero value on $x \in \mathrm{argmax}_x\, r(x)$ and zero value elsewhere:

$$g^*(x) \begin{cases} \neq 0 & \text{only if } \eta(x) = 1/2; \\ = 0 & \text{otherwise.} \end{cases} \tag{11}$$

And

$$\begin{aligned} \mathbb{E}_{x\sim g^*(x)} \left[r(x)\right] &= \int_{x\in \mathrm{supp}(g^*)} g^*(x) r(x) dx & (12)\\ &= \int_{x\in \mathrm{supp}(g^*)} g^*(x) \min\left(\frac{1}{2}, 1-\frac{1}{2}\right) dx & (13)\\ &= \frac{1}{2}\int_{x\in \mathrm{supp}(g^*)} g^*(x) dx & (14)\\ &= \frac{1}{2} \ . & (15) \end{aligned}$$

g^* does not need to be defined outside of $\mathrm{supp}(g)$. By playing g^*, G achieves global maximum $1/2$, which is the value of the game. □

[1] $\mathrm{supp}(g)$ is the *support* of distribution g, i.e., $\forall x \in \mathrm{supp}(g), g(x) > 0$.

The above analysis shows that **at equilibrium, both G and D finds the Bayes classifier, the optimal classifier for zero-one loss**. The generator draws training data on the Bayes decision boundary, where the discriminator performs random guess with 1/2 error rate. Note that the optimal solution for g is not unique (it can concentrate on arbitrary subset of points on the Bayes decision boundary), so the game has multiple equilibria with the same value.

3 Learning algorithms: finding equilibrium via computation

Discriminative model training algorithms are essentially computational approaches to finding the Nash equilibrium of the game. That is, we design G and D as algorithms, and let them play against each other. In this section, we show that both supervised learning and active learning algorithms can be viewed in this way; the difference is whether there is a human in the loop (providing labels on demand of G).

Note that in practice, we can only hope to approximate the equilibrium, since we only have (1) limited labeled data to estimate the expectations; (2) model class with limited capacity; (3) limited computational power to play G and D for finite number of iterations.

3.1 Supervised learning algorithms

The minimax game-play is the training process of quite a few supervised learning algorithms. In Table 1, we bring together notations of these algorithms.

(1) The **boosting** setting is long being connected to a minimax game [4]. The generator is the data reweighting scheme in each round, which puts higher and higher weights on hard examples around decision boundary; the discriminator is the ensemble of weak classifiers.

(2) In **SVM** learning, the generator is the set of positive Lagrange dual variables for each training example, which can be seen as unnormalized probabilities and will end up being positive only on support vectors (data points close to the boundary); the discriminator is the SVM classifier. The primal-dual convex optimization process can be interpreted as a minimax game [1].

(3) In **online convex optimization (SGD)**, the generator is the source of labeled data stream, while the discriminator is the classifier being updated. The online setting has several distinguishing features: the generator draws one example at a time, and discriminator tries to minimize cumulative regret instead of expected error. The connection between linear SGD and minimax game has been well studied [16].

Table 1: Game-theoretic view of supervised learning algorithms

Generator G	Discriminator D
Boosting	
$R_t(g_t, h_t) = \sum_i g_t(x_i)\mathbf{1}_{\{h_t(x_i)\neq y_i\}}$	
$g_{t+1}(x_i) \propto g_t(x_i)\exp\left(\alpha_t\left(2\cdot\mathbf{1}_{\{h_t(x_i)\neq y_i\}}\right)-1\right)$	$h_{t+1}(x) = h_t(x) + \alpha_{t+1}f_{t+1}(x)$
where $\alpha_t = \frac{1}{2}\ln\left(\frac{1-R_t}{R_t}\right)$	where f_{t+1} has error $R_{t+1} < 1/2$ on g_{t+1}
SVM	
$R(g,h) = \sum_i \alpha_i \ell(h(x_i), y_i) + \frac{1}{2}\left\|h\right\|^2$ (Lagrangian function)	
$g(x_i) \propto \alpha_i > 0$	$h(x) = w^\top x = \sum_i \alpha_i y_i k(x_i, x)$
where α_i's are dual variables	where w is a vector of primal variables
SGD	
$R(g,h) = \sum_t \ell(h_t(x_t), y_t) - \ell(h^*(x_t), y_t)$ (cumulative regret)	
$g_t(x) = 1/\left\|\mathcal{X}\right\|$; uniform	$h_{t+1}(x) = \text{argmin}_h \sum_t \ell(h_t(x_t), y_t) + \eta_t \left\|h\right\|^2$

3.2 Active learning algorithms

The minimax game also reminds us the active learning process: the generator G samples unlabeled data, and ask for labels; the discriminator then relearns a classifier from the labeled (and sometimes unlabeled) data. The sampling distribution g is sometimes referred to as the design distribution [14] in optimal experiment design. If the generator is only allowed to select existing data points from a pool, it is called pool-based active learning. If the generator is allowed to synthesize unseen data examples (as in the original GAN paper [7]), it is called membership query synthesis.

Conceptually, G should sample unlabeled data examples that the current classifier is uncertain about. The major difference between supervised learning settings (boosting/SVM/SGD) and the active learning setting is whether the training data are fully labeled before learning starts. Supervised learning settings assume that we have already collected many labeled data before learning, so that the generator can know which 'moves' are 'good' (which data distribution would incur high error rate on the discriminator), while the generator in active learning cannot know for sure except taking a guess.

Uncertainty sampling strategies can be understood as selecting the next unlabeled data point that *maximizes the guessed loss* of the current classifier. This is a greedy strategy. Since the guessed loss can be very inaccurate especially at the beginning, the selected data can be suboptimal.

Table 2: Uncertainty sampling algorithms: pick $x = \text{argmax}_x\, q(x)$.

Strategy	criteria $q(x)$	Equivalent loss	
least confident	$1 - p_h(y^*\|x)$	$\mathbb{E}_{p_h(y\|x)}\left[\mathbf{1}_{\{y^* \neq y\}}\right]$	zero-one loss
margin	$p_h(y'\|x) - p_h(y^*\|x)$	$\max(0, 1 + \max_{y \neq y^*} p_h(y\|x) - p_h(y^*\|x))$	hinge loss
entropy	$-\sum_y p_h(y\|x) \log p_h(y\|x)$	$\mathbb{E}_{p_h(y\|x)}\left[-\log p_h(y\|x)\right]$	log loss

Look-ahead strategies, such as expected error reduction and expected model change, avoid greedy selections by looking one step ahead. They look ahead by using the current guessed label *as is*, and retrain h for each 'pseudo-labeled' data. By looking ahead, these strategies outperforms the myopic uncertainty sampling, but they are also computationally expensive.

Variance reduction strategies implicitly look ahead. In bias-variance decomposition of expected loss, the variance term $\mathbb{E}_D\left[(h_D(x) - \mathbb{E}_D\left[h_D(x)\right])^2\right]$ depends on the training data D. Variance reduction strategies choose D such that the variance is maximally reduced after retraining. The variance can be either variance in model parameters (Fisher information matrix algorithms; A-optimality) or model outputs (query-by-committee; D-optimality). They are not as computationally demanding as explicit look-ahead strategies.

Minimax formulation has been used to analyze active learning in theoretical work [2, 8]. Reinforcement learning algorithms (multi-armed bandits [5]), submodular maximization [6], and max-min formulation [9], are used to solve active learning. But the literature did not explicitly connect active learning to a zero-sum game with two players as we do here.

4 Regularized Loss Maximization

In this section, we aim to design a data selection objective for G that is both principled and computationally inexpensive. The main idea is that if both G and D play optimally in each iteration, then they should converge to equilibrium with a fast rate. (If they were given infinite data and computation, they could converge in one step.) This is known as the best response dynamics, which is a straightforward way to find a Nash equilibrium [13].

In each iteration, D's strategy is to train a classifier that minimizes the *regularized* empirical loss:

$$h \leftarrow \underset{h \in \mathcal{H}}{\text{argmin}} \sum_{i=1}^{|S|} \ell(h(x_i), y_i) + \lambda \Omega(h) \ . \tag{16}$$

The regularizer $\Omega(h)$ puts preference on the classifier h, such as being less complex. By trading variance for bias, it can effectively reduce overfitting when the training data S is small. As S gets larger, D can decrease λ to obtain a consistent h. This is well-studied in structural risk minimization.

What does it mean for G to play *optimally*? Inspired by D's strategy, we can design G's strategy in a symmetric way: it aims to *maximize* the *regularized* empirical loss:

$$S \leftarrow \underset{S \subset X}{\text{argmax}} \sum_{i=1}^{|S|} \ell(h(x_i), y_i) + \mu \Phi(S) \ . \tag{17}$$

The regularizer $\Phi(S)$ puts preference on the sample S. Without the regularizer, Eq. (17) reduces to uncertainty sampling, where the greedy behavior is analogous to overfitting. It is sensible to design $\Phi(S)$

to prefer representative S, so that S does not focus on outliers. As S gets larger, G can decrease μ to concentrate S more on uncertain regions, eventually on the decision boundary.

We call this strategy **regularized loss maximization**. It is based on a game-theoretic framework, and it naturally extends to batch-mode active learning. It is different from the information density framework [15], because the selection criterion is now a set function instead of a pointwise ranking function. It is also connected to filtered active submodular selection [19], but our approach combines uncertainty sampling and submodular data selection in a more principled manner.

4.1 Regularizer design

The regularizer $\Phi(S)$ serves as a prior when empirical loss estimation is noisy due to an inaccurate classifier h. In such cases, $\Phi(S)$ should encourage S to *explore* the input space, i.e. to cover dense and diverse areas of the input space.

A function family that is well-suited for this purpose is the sensor placement functions in submodular maximization literature [10], such as coverage function, facility location, and mutual information. They all prefer representative and diverse sets, and efficient (greedy) maximization solution is proven to be near-optimal. Furthermore, submodular functions gracefully handle the cold-start problem: at the beginning of the game when we have no labeled data to train h, $\Phi(S)$ will guide G to query data points that are in dense and diverse areas of the input space.

(1) Coverage function: for every data point $x \in \mathcal{X}$, its neighborhood is a set of points $\Gamma(x) \subset \mathcal{X}$, including itself. Cover function is the sum of weights of every point in S and their neighbors:

$$\Phi(S) = \sum_{e \in \cup_{x \in S} \Gamma(x)} w(e) \ . \tag{18}$$

When $w(e) = 1$, it counts the number of elements covered by S. This function is useful for graph data representation but less so for vector data representation as it is hard to define rigid neighborhoods.

(2) Facility location: let $s(\cdot, \cdot)$ measure the similarity between data points. Facility location measure the similarity between S and unselected data U:

$$\Phi(S) = \sum_{e \in U} \max_{x \in S} s(e, x) \ . \tag{19}$$

(3) Mutual information: suppose we have a joint probability distribution of values on S and unselected data U. The mutual information $I(S, U) = H(U) - H(U|S)$. It can be modeled by the Gaussian processes [11]:

$$\Phi(S) = \frac{1}{2} \log\det\left(I + \sigma^{-2} K_S\right) \tag{20}$$

where K is a Gram matrix of elements $k_{i,j} = k(x_i, x_j), \forall x_i, x_j \in S$. Since $\log\det(\cdot)$ is concave, we can view this problem as selecting columns in the full kernel matrix K_D that are aligned with top eigenvectors, which is similar to cluster centers.

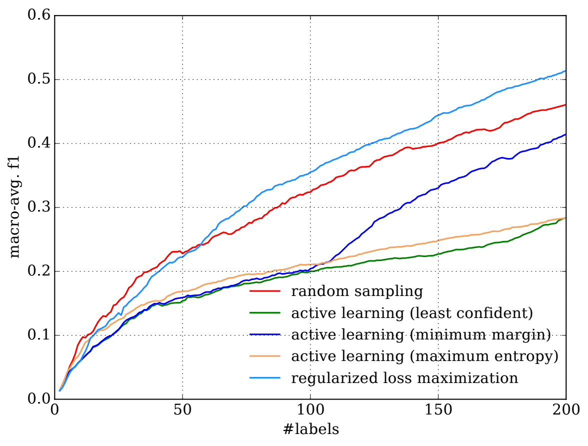
In practice, the regularization strength μ should decrease over time. In our plot study, we find that $\mu_0/|S|^\beta$ works well with $\beta = 1$ or 2 on text classification problems.

4.2 Theoretical justification in Bayesian optimization

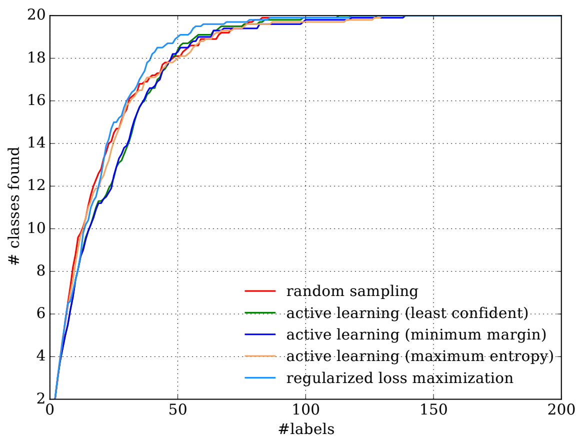
Why do we need a submodular regularizer in Eq. (17)? It can be understood as the exploration term in Bayesian global optimization algorithms.

In the limit of the game, we want to identify the region(s) in the input space where the expected loss achieves the maximum (1/2), but with finite samples our measurements of the loss (through h) is a noisy realization of the true risk function. Finding the region with the highest expected loss is similar to the global optimization problem, where we want to identify the input point(s) with the highest function value.

A theoretically sound solution to Bayesian optimization is multi-armed bandits algorithms, such as upper confidence bounds (UCB). It adopts the ‘optimism in the face of uncertainty’ principle: we pull an arm with the highest upper confidence bound. Note that here our ‘arms’ (input data points) are in a



(a) Macro-averaged F1 over 20 classes.



(b) Number of classes found.

Figure 1: Learning curves on the `20NewsGroup` data set. (a) Three simple uncertainty sampling methods underperform random sampling. In contrast, regularized loss maximization outperforms random sampling by an increasing large margin after querying 50 labels. (b) Regularized loss maximization discovers new classes faster than other methods, because of its balanced exploration-exploitation strategy.

metric space defined by features, or more generally, by a kernel function, so pulling one arm will reveal rewards of many other arms nearby. To model this phenomenon, a Gaussian process (GP) is used to model the unknown function (which corresponds to our loss function), where the correlation between 'arms' is modeled by a kernel function. This leads to the GP-UCB algorithm [17] that combines GP and UCB to identify the maximum points of an unknown function with minimal number of trials. In GP-UCB, the exploitation part is the mean, which corresponds to our empirical loss term; **the exploration part is the mutual information between the selected points S and the unselected points U**, which corresponds to our submodular regularizer $\Phi(S)$.

In principle, one can try out other Bayesian optimization approaches as G's sampling strategy, such as expected improvement. This is out of the scope of this paper.

4.3 Real data experiments

To illustrate the advantage of regularized loss maximization, we conduct experiments on the `20NewsGroup` dataset, where simple uncertainty sampling methods are known to underperform random sampling.

We implement regularized loss maximization using *hinge loss* of the multiclass classifier (Table 2, 2nd row). The regularizer is the *facility location* function (19), with $\mu = 1/|S|^2$. The learning curves are shown in Figure 1. The 20 classes in `20NewsGroup` dataset has non-trivial overlapping, i.e. label noise. Simple uncertainty sampling methods suffer in such case because the queried examples may be noisy boundary cases that are not representative enough to carry useful information for discerning the classes. Regularized loss minimization strikes a balance between exploration (querying representative examples) and exploitation (querying uncertain examples), outperforming random sampling. A closer look at the number of classes found by each algorithm shows that regularized loss maximization is the fastest in discovering new classes, as a result of explicit exploration.

5 Implication on interactive learning algorithm design

In an interactive learning setting, we want to explicitly design the generator G such that the game reaches a high-quality equilibrium as fast as possible.

First, we want the learned decision boundary to be complete. A poor equilibrium is that both g and h focuses only on a single point on the Bayes decision boundary which cannot generalize to other area of the data space. This problem can be alleviated by: (1) having a limited-capacity distribution class $\mathcal{G}$ and function class $\mathcal{H}$, so that g and h cannot concentrate (overfit) local area (linear models, regularized kernel machines, etc); (2) we don't allow sampling with replacement; (3) retrieving as diverse data as possible to cover the data space, so that (at least) the Bayes decision boundary is surrounded by data examples (**ReQuery** [12]).

Second, a 'knowledgeable' generator can accelerate learning. (1) The generator may know that some region in X-marginal data distribution have high density, so sampling data from that region can probably incur high error (density-based sampling). (2) The generator may already know some direction that are orthogonal to the decision boundary (**informative features**), and data points along that direction most likely pass through 'hard' data points on the decision boundary. A direction parallel to the decision boundary is not likely to contain 'hard' data points. It is good to have multiple uncorrelated directions that are all orthogonal to the decision boundary (**diverse informative features/queries**), so that when we retrieve many 'hard' data points along different directions. (3) The generator may have acquired knowledge from other related tasks (**multi-task learning**), so the hard data points in related tasks may as well be the hard data points in the current task.

Third, we need sufficient samples where the labels are noisy. If the labels are noisy and too few, it is possible to have a training set that lead to opposite classification, e.g. the set contains only negative examples on the positive side, and positive examples on the negative side. In the theoretical analysis, such scenario does not happen because the analysis holds *in expectation*, where enough examples should have been drawn from the noisy area.

References

[1] Stephen Boyd and Lieven Vandenberghe. *Convex optimization, Chapter 5 "Duality"*. Cambridge university press, 2004.

[2] Rui M Castro and Robert D Nowak. Minimax bounds for active learning. *IEEE Transactions on Information Theory*, 54(5):2339–2353, 2008.

[3] Gregory Druck, Gideon Mann, and Andrew McCallum. Learning from labeled features using generalized expectation criteria. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 595–602. ACM, 2008.

[4] Yoav Freund and Robert E Schapire. Game theory, on-line prediction and boosting. In *Proceedings of the ninth annual conference on Computational learning theory*, pages 325–332. ACM, 1996.

[5] Ravi Ganti and Alexander G Gray. Building bridges: viewing active learning from the multi-armed bandit lens. *arXiv preprint arXiv:1309.6830*, 2013.

[6] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[8] Steve Hanneke and Liu Yang. Minimax analysis of active learning. *Journal of Machine Learning Research*, 16(12):3487–3602, 2015.

[9] Steven CH Hoi, Rong Jin, Jianke Zhu, and Michael R Lyu. Semisupervised svm batch mode active learning with applications to image retrieval. *ACM Transactions on Information Systems (TOIS)*, 27(3):16, 2009.

[10] Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability: Practical Approaches to Hard Problems*, 3(19):8, 2012.

[11] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9(Feb):235–284, 2008.

[12] Cheng Li, Yue Wang, Paul Resnick, and Qiaozhu Mei. Req-rec: High recall retrieval with query pooling and interactive classification. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*, pages 163–172. ACM, 2014.

[13] Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*, volume 1. Cambridge University Press Cambridge, 2007.

[14] Maxim Raginsky and Alexander Rakhlin. Lower bounds for passive and active learning. In *Advances in Neural Information Processing Systems*, pages 1026–1034, 2011.

[15] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1070–1079. Association for Computational Linguistics, 2008.

[16] Shai Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

[17] Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

[18] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. Irgan: A minimax game for unifying generative and discriminative information retrieval models. *arXiv preprint arXiv:1705.10513*, 2017.

[19] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1954–1963, 2015.