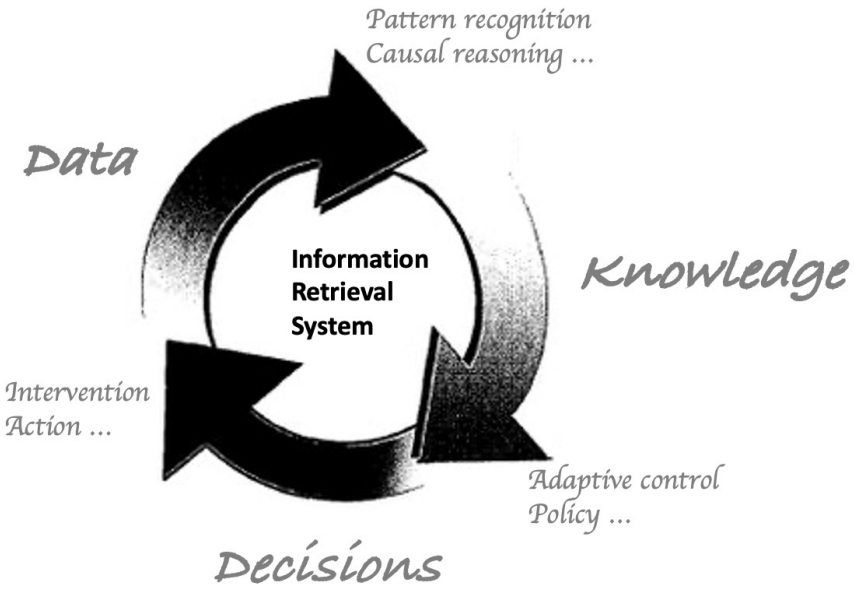
Theoretical Foundation for Modern Information Retrieval System

KDD 2022 Tutorial

Presenter: ***Da Xu***

Aug. 14, 2022



Contact: daxu5180@gmail.com (daxu5180.github.io)
Website: theoreticalfoundation4irsystem.github.io/Tutorial-KDD22

What this tutorial is about?

- Some observations and motivations:
  - Problem solving = f(data, deep learning, Hail Mary)?
  - Innovation in IR = rephrasing ideas/models developed elsewhere?
  - IR system = $\sum functions$?

- I know all about afternoon sessions and lunch coma, so:
  - Max intuition, minimum math!
  - A broad range of topics will be covered for the diversity of story telling.

- After 4 hours, hopefully, you will:
  - Gain a rough idea about: What makes an IR system system?
  - Capture the key insights and challenges behind IR's pattern recognition, causal inference, decision making, and system design.

Historical notes about IR

-- what makes IR the way it is today

- *Origins of IR*
  - **1920s**: "... Search for documents stored on microfilm ... ",
  - **1940s**: ".. US military indexing and retrieve documents captured from German ... "
- *Cranfield experiments, Lockheed system*
  - **1960s**: "exact retrieval", indexing, compressing, abstracting ...
  - **1970s**: first large-scale retrieval system
- *ACM SIGIR (1978 -)*
  - **1980s**: vector space model, clustering, naive Bayes, probabilistic model
  - **1990s**: search engines, graph link analysis, learning to rank, ...
  - 21st century: media search, recommender system, document classification, spam filtering, question answering, personalization

What makes IR a special discipline?

- **Intervenable** vs. **Observational**
  - Ability to take actions and interact with the world?

- **Generative** vs. **Discriminative**
  - Knowledge about data generation useful to the task?

- **Evaluative** vs. **Instructive**
  - Interpretate collected feedback as evaluation or instruction?

Introduction – Theoretical Foundation for IR Sys

Da Xu
ML Manager, Staff ML Eng
Search & Recommendation
Walmart Labs

Part 1: Pattern Recognition

- *ML Basics*:
  - Expressivity, Optimization, Generalization
- *Understanding Deep Learning*:
  - Double descent in bias-variance tradeoff
  - The kernel regimes
  - Implicit regularization
  - Generalization, memorization, and subpopulation
  - Role of model architecture
- *From classification to ranking*
  - Consistency / generalization of ranking
  - On smooth ranking loss
- *Domain challenges of IR data*:
  - Unlabellness: transductive (semi-supervised), domain adaptation
  - Sparsity: representation learning, graph neural network
  - Debias, extrapolation

Introduction – Theoretical Foundation for IR Sys

Part 2: Intervention and Causal Inference

- *The causal language*
  - Pearl and Rubin causal model
  - Invariant mechanism, confounding, randomization, counterfactual
- *Design and inference*
  - A/B testing, metric, and continuous monitoring
  - Best-arm identification and sequential testing
  - Interleaving and dueling
- *Observational studies and offline learning*
  - Is observational studies a missing data problem?
  - Double learning and targeted maximum likelihood learning
  - Propensity weighting method and counterfactual learning
  - Multiple causes, deconfounding, robust optimization
- *Conection to IR pattern recognition*
  - Causality and learning
  - Conformal prediction, learn-then-test

Da Xu
ML Manager, Staff ML Eng
Search & Recommendation
Walmart Labs

Introduction – Theoretical Foundation for IR Sys

Part 3: Action and sequential decision making

- *Online learning with evaluate feedback*:
  - Multi-armed bandit and exploration-exploitation
  - Dynamic system, planning, Markov decision process
- *The power of structure*:
  - Benefiting from structured policy & reward
- *Policy learning*
  - The hateful horizen & distribution shift
  - Supervised / representation learning challenged
  - Gradient or random search -- zeroth-order optimization
  - Model-based vs. model-free

Part 4: System design

- *Interaction between system components*
- *Pretrained embedding, negative sampling, calibration*
- *Decentralized system*

Da Xu

ML Manager, Staff ML Eng

Search & Recommendation

Walmart Labs

Bo Yang

ML Engineer

LinkedIn Ads AI

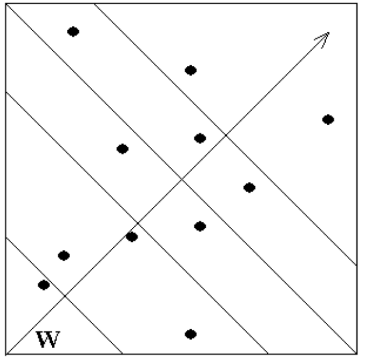
Just a few things before rock-n-roll …

- Three sessions + two breaks
  - 1st break at 2:00 pm.
  - 2nd break at 3:00 pm.
- Q&A and discussion
  - During the break
  - Approaching the end
- Very important: your feedback and comments!

Contact: daxu5180@gmail.com (daxu5180.github.io)
Website: theoreticalfoundation4irsystem.github.io/Tutorial-KDD22

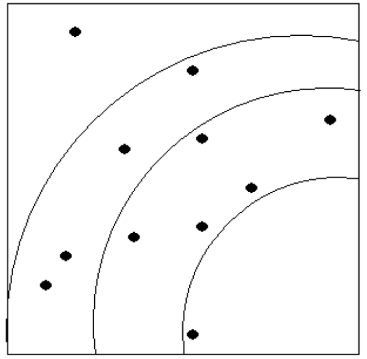
Part 1 – Pattern Recognition with Feedback Data

- Given a user query (or hidden intention) represented by X_u, the system reveals documents (items) each represented by X_i, and the user provide feedback $Y_{u,i}$



- Does it make a difference $Y_{u,i} \in \{0, 1\}$ or $Y_{u,i} \in \{0, 1, 2, \ldots, 5\}$?
  - Discretized response surface vs. partition boundary?
  - Partitioning boundary vs. decision boundary?
  - Will come back to this later ...

- Collected feedback data $\{(X_u, X_i, Y_{u,i}) \mid (u, i) \in D\}$



- Employ $f \in \mathcal{F}$ to learn the patterns s.t. $f_{u,i} := f(X_u, X_i)$ predicts preference / how likely to click.

A 10-min crash course on ML basics

- *Estimation* and *prediction* are central to statistics and ML.
- We start with the former which does not involve relationship among variables:
  - Independent random variable $X_1, \ldots, X_n$ sampled from the same distribution, assuming standard Gaussian or *1-subgaussian*
  - Often interested in the mean $\mu := \mathbb{E}[X_i]$ -- when is the sample average a good estimation?
  - Let $\hat{\mu} = 1/n \sum_{i=1}^{n} X_i$, then according to *Chebyshev's inequality*:

$$p(\hat{\mu} \geq \mu + \epsilon) \leq \exp(-n\epsilon^2/2), \forall \epsilon > 0$$

  - The above *concentration* result holds similarly in the other direction. They imply:

$$\mu \in \left[\hat{\mu} - \sqrt{\frac{2\log(1/\delta)}{n}}, \hat{\mu} + \sqrt{\frac{2\log(1/\delta)}{n}}\right], \quad \text{with probability at least } 1-\delta$$

  - The same $1/\sqrt{n}$ rate as *Law of Large Numbers*.

A 10-min crash course on ML basics (classification)

- Prediction with a *full knowledge* about the population distribution $p(X, Y)$
  - Via optimization using loss function and risk $R(f) := \mathbb{E}[\ell(f(X), Y)]$
  - The optimal predictor follows the likelihood ratio test, $f^*(x) = \mathbb{1}\left[\frac{p(X = x|Y = 1)}{p(X = x|Y = 0)} \geq \gamma\right]$ where the threshold depends on the 0-1 loss
  - Using Bayes rule, it is equivalent to $f^*_{Bayes}(x) = \arg\max_{y \in \{0,1\}} p(Y = y|X = x)$
  - Its optimality <- *Neyman-Pearson lemma*.
- Prediction with *i.i.d random samples* $(x_1, y_1), \ldots, (x_n, y_n) \sim p(X, Y)$
  - "Supervision" – availability of the label
  - Proper loss function -- Bayes consistent in the sense that: $\mathbb{1}[f^*_\ell(x) > 0] = f^*_{Bayes}(x)$ (exponential loss, logistic loss, ...)
  - Optimizing empirical risk: $R_n(f) := 1/n \sum \ell(f(x_i), y_i)$
  - Why does it work? $R(f) = R_n(f) + (R(f) - R_n(f))$

A 10-min crash course on ML Bascis

- Take a look at $R(f) = R_n(f) + (R(f) - R_n(f))$:
  - Since we are searching within $f \in \mathcal{F}$, is the *hypothesis class* expressive enough to represent the inductive bias of the data and task? (**Expressivity**)
  - How to find the hypothesis with: $\arg\min R_n(f)$? (**Optimization**)
  - How large could $R(f) - R_n(f)$ possibly be? (**Generalization**)

- Understanding the *expressivity* of $f_\theta(x), \theta \in \Theta, x \in \mathcal{X}$:
  - Structure of the *feature space* – vector space, string, graph, kernel, basis functions, etc
  - Structure of the *model space* – linear, sequential, modular, convolution, etc
  - Parameterization and smoothness

- Optimization for $R(\theta) := R(f_\theta)$, assuming convexity:
  - "Descent direction" – $R(\theta + \alpha v) < R(\theta), \alpha > 0$
  - Characterize descent direction – *negative gradient* (under good condition), because

$$v = \nabla R(\theta) \Rightarrow R(\theta + \alpha v) < R(\theta)$$

A 10-min crash course on ML Basics

- Gradient descent optimization:
  - Iterative update $\theta_{t+1} = \theta_t - \alpha_t \nabla R(\theta_t)$
  - Usually ends at *stationary point* where gradient vanishes $\theta^* : \nabla R(\theta^*) = 0$
  - Stochastic GD – evaluate gradient at a random sample, gradient is unbiased

    $$g(\theta; i) = \nabla \ell(f_\theta(x_i), y_i);\ \mathbb{E}[g(\theta; i)] = \nabla R(\theta)$$

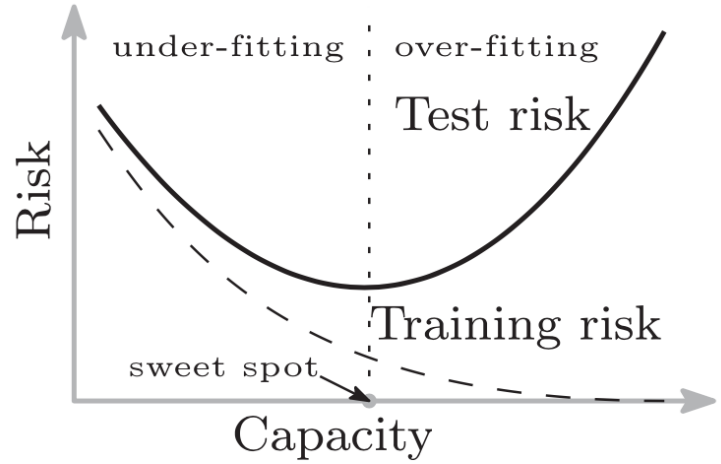
  - When and Why does SGD work?

    If the gradient's variance $\mathbb{E}[\|g(\theta; i)\|^2]$ is bounded, then the accumulated error due to the randominess still converges at a $\sqrt{T}$ rate, i.e. $\mathbb{E}[R(\bar{\theta}) - R(\theta^*)] = \mathcal{O}(1/\sqrt{T})$

- Generalization of empirical risk minimization:
  - Recall $R(f) = R_n(f) + (R(f) - R_n(f))$ -- optimization makes the 1st term small, the 2nd term is *generalization gap*
  - Generalization gap has been shown to related with *model complexity*, *stability*, and *margin*

A 10-min crash course on ML Basics

- The most well-known criteria for *generalization gap* is *model complexity*
  - *VC dim:* abiliy to conform to an arbitary labeling of data points
  - *Rademarcher complexity:* ability to interpolrate random sign pattern
  - *"Uniform convergence":* applies to all functions in $\mathcal{F}$



For *L-Lipschitz loss*: $R(f) - R_n(f) \leq 2L\mathcal{R}_n(\mathcal{F}) + 3\sqrt{\frac{\log 1/\delta}{n}}, \forall f \in \mathcal{F}$

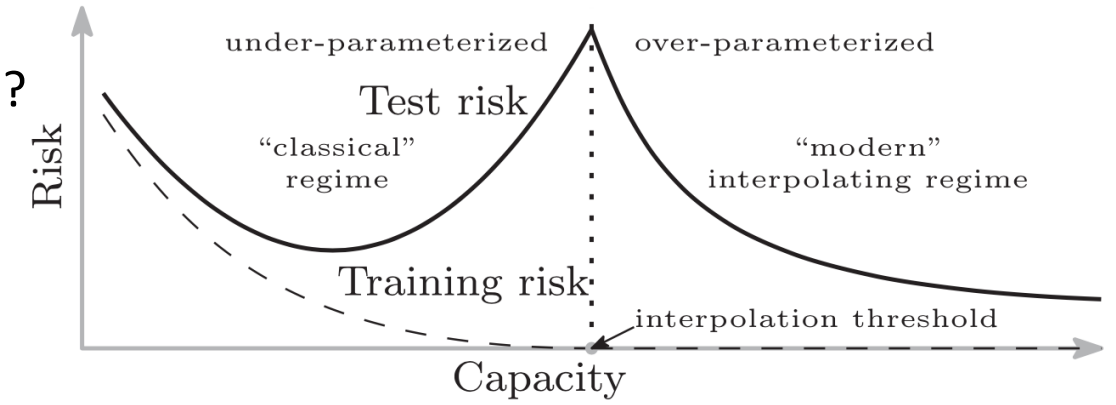
- *Margin-based* generalization bound
  - *Margin* is also a form of regularization (why?) that improves generalization
  - Let $R_n^\gamma(f)$ be the empirical risk wrt. the margin loss $\mathbb{1}[yf(x) \leq \gamma]$: $R(f) - R_n^\gamma(f) \lesssim \frac{\mathcal{R}(\mathcal{F})}{\gamma} + \sqrt{\frac{\log 1/\delta}{n}}$

- *Algorithmic-stability* bound
  - *Sensitivity* to the change of a single training sample
  - If *insensitive* to such perturbation, then generalization gap should be small. (Will revisit this in more detail.)

Understanding deep learning

- When classical bias-variance phenomenon fail
  - The *double descent* phenomenon
  - Finding the best inductive bias and global optimum?
  - Rethinking uniform generalization.
- The *limiting kernel* regimes
  - On the expressivity of overparameterized NN
  - The *optimization path* under GD
  - *Neural tangent kernel*
  - Adaptive feature learning
  - On the complexity of NN *

[BHM+19]



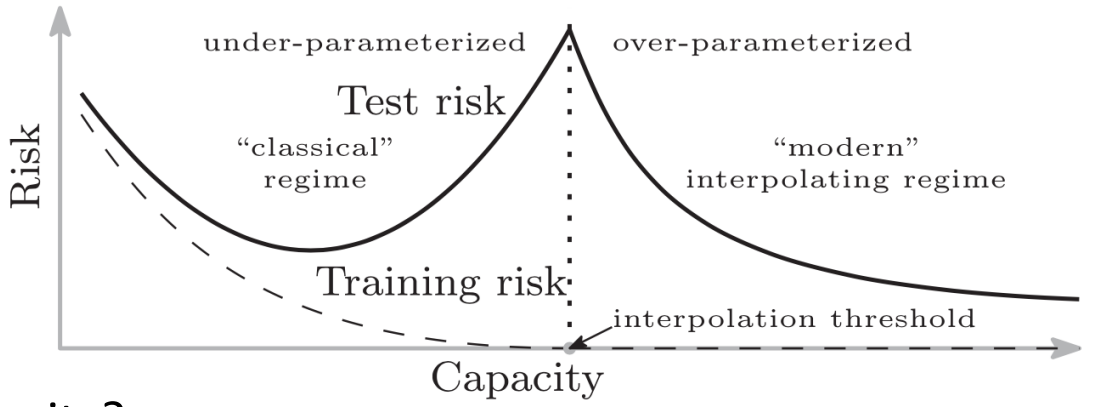
* we will use $\mathcal{C}(\mathcal{F})$ as a general complexity measure, which could be the previous Rademarcher complexity, naive count, VC dimension, covering number, etc

Understanding deep learning

- *Implicit regularization* of overparameterized models
  - Silver lining for non-convex ERM
  - Does SGD randomly pick a local optimum?
  - The role of *minimum-norm solution*
- Current understanding of how NN generalize:
  - A combined story of overparameterization, implicit regularization, and kernel
  - Through the lens of *memorization*
  - The under-investigated *fairness for subpopulation*
- The role of *structure* for IR models
  - Efficient optimization -- rethinking *auto-differentiation and backpropagation*
  - Inductive bias and the *smoothness of interpolating hypothesis*

Understanding deep learning

- Before the double descent phenomenon:
  - Overparameterized DL models routinely achieve almost-zero training risk
  - Desipte the near-perfect fitting and very high model class complexity, they still generalize well, even under high label noise
  - The sweet spot between underfitting and overfitting is gone?
- Beyond the *interpolation point*
  - Modern practice (d>n) is shown to the right of interpolation point
  - Training and testing risk decrease simultaneously
  - What is the inductive bias that perfectly describes this phenomenon?
  - How does SGD find a good solution depite non-convexity?
  - Can uniform convergence still explain NN's generalization?



Let's start with some intuition

- Guessing the general inductive bias for overparameterized NN models
  - Uniform approximation theorem?

    -- Capacity of functional class does not necessarily reflect how well the predictor matches the problem at hand ...
  - Represent many decision boundaries that perfectly separate data + SGD find the smoothest one?

    -- Seems like a form of *Occam's razor*: the simplest explanation compatible with observations
- But still huge gaps with current understanding, e.g.
  - 1). why can we expect the bahavior of SGD for non-convex function;
  - 2). interpolating the data means complexity grows with data (especially for *noisy data* cases), why can we expect any generation from the solution?
- Optimization-wise, could it be over-parameterization give rise to some *tractability*?
- Generalization-wise, could it be over-parameterization and SGD optimization to find a *small-norm* solution (corresponding to smoothness) + high-capacity component?

Gradient decent in the over-parameterized regime

- A change of direction – from *convergence of parameter* to *convergence of prediction*
  - Regardless of the parameter space, empirical evidence suggests the predictions can constantly reach zero training risk
  - Consider *square loss* for simplicity: $(f_\theta(x_i) - y_i)^2$, and $f_\theta(x) = \theta^T x$
  - First note that under GD update, it holds:

$$\hat{y}_{t+1} - y = (I - \alpha X X^T)(\hat{y}_t - y)$$

  - So, if α is small and XX^T is strictly positive definite, then predictions converge to training labels
  - A necessary condition for that is $d > n$.
  - Let's move on to non-convex models. Using first-order Taylor expansion, under GD update:

$$\hat{y}_{t+1} - y = (I - \alpha \mathcal{J} f_\theta(x)^T \mathcal{J} f_\theta(x))(\hat{y}_t - y) + \alpha \epsilon_t$$

    where $\mathcal{J} f$ is the Jacobian (because x, y are now the vectorized collections), and ϵ_t is second-order residue (usually small).
  - So non-convexity not really matter here if step size is small – but the Jacobian is again likely to be positive definite if $d > n$.

Gradient decent in the over-parameterized regime

- Just as it seems non-convexity doesn't really get in the way for *prediction's convergence under over-parameterization*, reaching *zero-loss* leads to something even better
  - because of zero-loss: $y = \theta^T x$
  - because GD/SGD: $\theta_{t+1} = \theta_t - \alpha e_t x_t$, e_t : gradient of the loss as current prediction
    the algoirthm searches over a space with dimension equal the number of data, and
    $\hat{\theta} = X^T \beta$ because we are in the span of the data (assume initialization at zero)
  - what is special about solutions of this form?
  - assume another solution, $\hat{\theta}' = X^T \beta + v,\ v \perp x_i$
  - easy to verify: $\|\hat{\theta}'\|_2 = \|\hat{\theta}\|_2 + \|v\|_2$ -- so $\hat{\theta} = X^T \beta$ is also the minimum-norm solution.
- Recall *kernel methods* and the *representer theorem*
  - consider L2 regularized ERM, then: $\hat{\theta}' = X^T \beta + v,\ v \perp x_i \rightarrow \frac{1}{n} \sum_i \ell(\beta^T X x_i, y_i) + \lambda \|X^T \beta\|^2 + \lambda \|v\|^2$
  - risk minimized at $v = 0$ -- optimial lies in the span of the data
  - can instead search for solution in the kernel expansion: $K = XX^T$

Gradient decent in the over-parameterized regime

- Minimum-norm solution and margin
  - Recall from SVM that if we correctly classify all data, then margin satisfies $\gamma(\theta) = \min_i \frac{y_i \theta^T x_i}{\|\theta\|}$
  - Suppose we interpolate the data: $y_i = \theta^T x_i$, then $\gamma(\theta) = \|\theta\|^{-1}$

    solution from SGD achieves interpolatiion and max-margin solution
- Moving to non-linear models
  - Similar convergence of prediction behavior:

$$\hat{y}_{t+1} - y = (I - \alpha \mathcal{J} f_{\theta^{(t)}}(x)^T \mathcal{J} f_{\theta^{(t)}}(x))(\hat{y}_t - y) + \alpha \epsilon_t,\ \epsilon_t = o(\alpha\kappa\|\hat{y}_t - y\|^2)$$

    where κ bounds the curvature.
  - Nonconvexity doesn't really stand in the way to follow the previous argument
  - Let's make it rigorous for classification methods, with
    - loss function has *exponential-tail* behavior
    - the predictor is *homogeneous* – $f(c \cdot \theta, x) = c^q f(\theta, x)$
    - some *smoothness and Lipschitzness* of the predictor

Gradient decent in the over-parameterized regime

- Implicit regularization of gradient descent
  - A difference with the squre loss is that gradient descent now tends to increase the norm of the parameters, i.e. $\lim_{t\to\infty} \|\theta^{(t)}\| = \infty$
  - Why? Intuitively, both the risk and gradient has the form: $\sum_i C_i \exp(-y_i f_\theta(x_i))$

    and since $f_\theta(x) = \|\theta\|^q f_{\theta/\|\theta\|}(x)$, it holds that: $\lim_{t\to\infty} \exp(-y_i f_{\theta^{(t)}}(x_i)) \to 0, \forall i$

    so we reach both zero-loss (*interpolation*) and zero-gradient (*stationarity*)
  - One further implictation: due to the exponential tail behavior, during optimization, the risk will be *dominated* by the sample with ***largest margin***: $\arg\max_i y_i f_{\theta^{(t)}}(x_i)$
  - They are like *support vectors* in SVM!
  - So the optimization path is similar to that of the hard-margin SVM:

$$\min \|\theta\|_2 \ \text{ s.t. } \ y_i f_\theta(x_i) \geq 1, \forall i$$

  - Under appropriate step size and separability, let $\hat{\theta}$ be the solution to the above *L2 max-margin problem*, then the optimzation path of GD follows: $\lim_{t\to\infty} \frac{\theta^{(t)}}{\|\theta^{(t)}\|} = \frac{\hat{\theta}}{\|\hat{\theta}\|}$

[SHN+18]

Margin, scale, and generalization of NN

- We just saw GD/SGD optimization carries the implict bias of favoring minimum-norm large-margin solution, which suggest for generalization:
  - The *critical role of* ***margin*** – where margin can often be treated as a form of ***generalization***
  - The ***size*** (scale) of the parameters matters perhaps more than the ***number of parameter***

    -- can this be a new direction for explaining the complexity of NN and grant compatible with uniform generalization?
  - Recall that for linear classifier, $\gamma(\theta) = \|\theta\|^{-1}$
- A margin bound for uniform generalization of NN
  - Margin is sensitive to scale, which depends on the norm of NN
  - With parameters layered up, defining a norm for NN is non-trivial
  - *Spectral norm / Forbenius norm* of each layer -> ***peeling*** -> *size independent complexity* of NN

    -- together with the *margin bound*, we finally reach:

[GRS19]

$$p_{\text{test}}(yf_\theta(x) \le 0) \le \frac{1}{n}\sum_i \mathbb{1}[y_i f_\theta(x_i) \le \gamma] + \frac{\sqrt{q}\sum_{j=1}^{q} M(q)}{\theta\sqrt{n}}, \; M(q) \text{ is the norm bound of layer q}$$

Implicit regularization and the kernel regime

- The previous result captures the essence of margin, but that's not the whole story
  - *Margin* might just be one form of implict regularization
  - The *norm bounds* can be loose in general
  - Equivalent notion exists for such as *square loss*?
  - remember in the linear regime, we connect the implicit bias of GD with the *representer theorem and kernel*
  - This connection holds for NN as well, under certain *limiting approximation*
  - an observation: many parameters (under random initialization) change very little during training, so –

$$\hat{y}_{t+1} - y = (I - \alpha \mathcal{J} f_{\theta^{(t)}}(x)^T \mathcal{J} f_{\theta^{(t)}}(x))(\hat{y}_t - y) + \alpha \epsilon_t$$

$$\Downarrow$$

$$\hat{y}_{t+1} - y = (I - \alpha \mathcal{J} f_{\theta^{(0)}}(x)^T \mathcal{J} f_{\theta^{(0)}}(x))(\hat{y}_t - y) + \alpha \epsilon'_t,$$

  - Define the ***Neural Tangent Kernel*** $K(x, x') = \mathbb{E}_{\theta^{(0)}}\left[\langle \nabla f_{\theta^{(0)}}(x), \nabla f_{\theta^{(0)}}(x') \rangle\right]$
    which depends only on the input, initialization and model structure

[JGH18]

The kernel regime of NN

- An alternative way to understand the connection between GD implicit bias and neural tangent kernel is as follows:
  - Directly conider the Taylor expansion at initialization:

$$f_\theta(x) = f_{\theta^{(0)}}(x) + \langle \nabla f_{\theta^{(0)}}(x), \theta - \theta^{(0)} \rangle + \epsilon$$

  - We can always use reparameterization to get rid of the intercept – $f_\theta(x) \approx \langle \nabla f_{\theta^{(0)}}(x), \theta - \theta^{(0)} \rangle$
  - In this way, $\nabla f_{\theta^{(0)}}(x)$ acts like a *feature lift ,* and the correspond RKHS is induced by the *neural tangent kernel*: $K(x, x') = \langle \nabla f_{\theta^{(0)}}(x), \nabla f_{\theta^{(0)}}(x') \rangle$
  - According to the previous derivation for *kernelized ERM*, it is easy to see the problem becomes:

$$\text{minimize}_\beta \frac{1}{n} \sum_i \ell(e_i^T K\beta, y_i) + \lambda \|\beta\|_K, \; e_i \text{ is the Euclidean basis vector}$$

  - Therefore, we will reach a *minimum RKHS-norm solution* defined by the *neural tangent kernel*, and small norm in RKHS generally suggests *smoothness*.
  - Note that the above results are *qualitative* and often don't reflect what happens in practice.

The kernel regime of NN

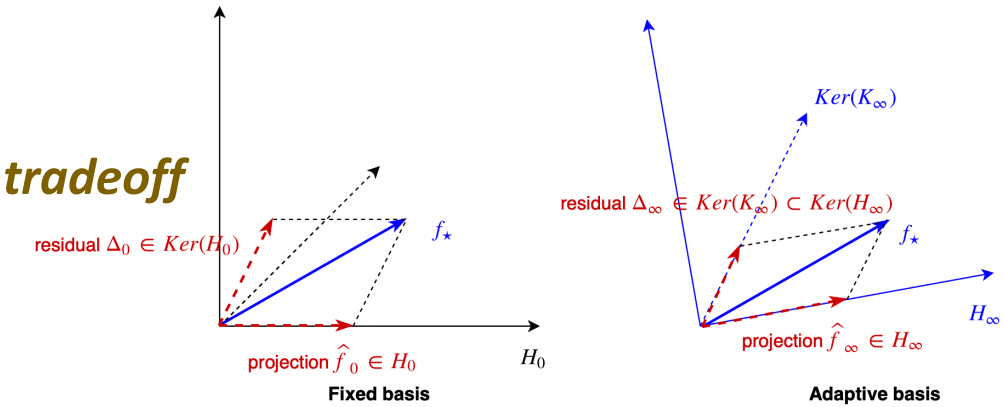
- What is the major gap between the NTK arugments and practice?
  - The NTK argument holds under *infinite-wide* setting and trained with *infitesemal learning rate* (second-order bias diminishes)
  - It resembles a *random-feature model* (e.g. the representations follows particular random initialization) with one trainable linear layer
  - In practice, (hidden) representations prior to the last linear layer can be updated during training
- Adaptive kernel (feature) learning of NN
  - The extra expressive capacity of NN creats a *decomposition* of the kernel space
  - One corresponds to the *NTK component*, the other is a *data-adaptive component*
  - Corresponds to decomsing the predictor into a *regular*

    prediction component (that generalizes well) and a

    interpolating component, they give ***different bias-variance tradeoff***

[GJK21]

$$\hat{f} = \hat{f}_{\text{pred}} + \hat{f}_{\text{interp}}$$



The role of model structure

- Containing more smooth decision boundaries that interpolates the data
  - IR data are extremely noisy
  - Interpolating IR dat often require both appropriate inductive bias and larger capacity
  - Smoother interpolation -> less sensitive to individual samples (thus better stability and generalization)
- Making GD suffering less from vanishing/exploding gradient (Jacobian more well-conditioned)
  - *Residual connection* – adding identity matrix
  - *Batch normalization* – similar scale across layers
  - *Dot product fusion* – spectral norm as regularization

Stability and memorization

- Recall that the *stability* of a learning algorithm also suggests *generalization*
  - NN can almost achieve interpolation even if we assign random label to a sample
  - Although directly deriving the stability of NN is challenging, empirically we know they are stable
  - Even with seemingly outlier samples from under-represented group, NN can fit those perfectly – is it a result of $\hat{f}_{\text{pred}}$ or $\hat{f}_{\text{interp}}$?
  - Why do we care? Data in IR is often a *mixture of individual distributions*, some users are well-represented, some are under-represented.
  - If samples from under-represented users are fitted because $\hat{f}_{\text{interp}}$ exploits ***supurious relationship*** (e.g. ***memorizing*** them using the model's extra capacity), then the testing performance will be bad on those users (***unfairness***)

[FZ20]

Memorization of NN and under-investigated fairness

- We've seen before that *over-parameterization* and ***interpolation*** are crucial to the success of NN – *defined by generalizing to the same population (distribution) seen by the training data*
  - ***Memorizing*** the under-represented subpopulations seems inevitable, judging both by the means and the goal
  - Does importance weighting address this issue?
- Understanding ***importance weighting*** for deep learning
  - When data is inseparable, importance weighting helps the training risk emphasizing the different sub-populations, e.g. with training data collected from P and we target at Q, the generalization bound is dominated by

[XYR21]

$$\mathcal{O}(\frac{D(P,Q)\mathcal{C}(\mathcal{F})}{\sqrt{n}}), \text{ where } D(P,Q) \text{ is the discrepancy between } P \text{ and } Q$$

  - Upon achieving interpolation, however, the convergence to the minimum-norm max-margin solution is *not altered by reweighting*!

$$\| \frac{\theta^{(t)}}{\|\theta^{(t)}\|} - \frac{\hat{\theta}}{\|\hat{\theta}\|} \| = o(\frac{\log n}{\log t})$$

From classification to learning-to-rank

- What we learned so far also applies to ranking problem?
  - Prediction with full knowledge about the distribution?
  - Generalization wrt. the decision rule? The rate?
- The ***ranking problem***
  - Given two pairs of $(X, Y), (X', Y')$, define $Z = \frac{Y - Y'}{2}$ -- in practice, we only care about its sign
  - A ***ranking rule*** is: $r : \mathcal{X} \times \mathcal{X} \to \{-1, 1\}$, and its performance is measured by the ***ranking risk***

$$R(r) = p(Z \cdot r(X, X') < 0)$$

  - If the whole distribution is known, easy to verify that the optimal rule also follows the max-a-poteriori principle

$$r^*(x, x') = 2\mathbb{1}\left[p(Z > 0 \,|\, x, x') \geq p(Z < 0 \,|\, x, x')\right] - 1$$

  - Now we move on to the ***empirical ranking risk***

$$R_n(r) = \frac{1}{n(n-1)} \sum_{i \neq j} \mathbb{1}[Z_{i,j} \cdot r(X_i, X_j) < 0]$$

From classification to learning-to-rank

- Ranking, scoring, and the bipartite problem
  - Would like to reduce ranking to scoring so we can unleash our zoo of ML methods
  - It is possible when the joint distribution is such that there exists f^* such that:

$$r^*(x, x') = 1 \text{ if and only if } f^*(x) \geq f^*(x')$$

  - So the risk becomes: $R(f) := 2\mathbb{1}[f(x) \geq f(x')] - 1$
  - A particular interesting case is $Y\ in\ \{-1, 1\}$, because we can show that

$$\mathrm{AUC}(f) = p(s(X) \geq s(X') \,|\, Y = 1, Y' = -1) = 1 - \frac{1}{2p(1-p)} R(f),\ p = p(Y = 1)$$

  - Hence minimizing ranking loss is ***optimizing the AUC***
- ***U-statistics*** and *improved generalization bound*
  - The ranking loss is a ***degree-2 U-statistics*** -- *minimum variance* among all unbiased estimators.
  - What we encountered so far are all based on *Hoeffding and Mcdiarmid's* inequality based on first-order statistics. There is also the ***Bernstein-type*** bound

$$R(f) - R_n(f) \leq \mathcal{O}\left(\sqrt{\frac{V_n(f)\log(\mathcal{C}(\mathcal{F})/\delta)}{n}}\right), \text{ with probability at leat } 1 - \delta$$

From classification to learning-to-rank

- The ***generalization bound*** for ranking loss is generally: $\mathcal{O}\left(\left(\frac{\mathcal{C}(\mathcal{F})\log(n/\delta)}{n}\right)^{1/(2-\alpha)}\right)$ [CLV08]
  - Seems like an improvement to standard supervised learning,
    if finding *appropriate formulation* that satisfies $\alpha > 0$
  - But it still comes to finding a surrogate loss and do optimization
- RANKBOOST – using convex surrogate loss function
  - Use such as *exponential function*, *logistic cost*, or *hinge loss*, e.g.

$$R(f) = \mathbb{E}\exp(-\mathrm{sgn}(Z)\cdot f(X, X'))$$

  - The structure of the risk is now more *supervised-learning alike*
  - Indeed, the generalization error bound now becomes a standard one

$$R(f) - R_n(f) \leq \mathcal{O}\left(\frac{\mathcal{C}_n(\mathcal{F})}{\sqrt{n}} + \sqrt{\frac{\log 1/\delta}{n}}\right) \qquad \text{[RCM05]}$$

From classification to learning-to-rank

- When using metrics other than *AUC*, the previous approach do not directly optimize the ranking metrics (e.g. *AP, NDCG*)
  - The main difficulty for directly optimizing ranking metrics is that they are *piecewise constant*
  - Use some smooth approximation to approximate ranking?
  - Given K numbers $f_1, \ldots, f_K$, assign a probability for each of them to being the largest (we will see this *softmax trick* again): $p_i := \frac{\exp(f_i/\eta)}{\sum_{j=1}^{K} \exp(f_i/\eta)}$ [CW10]
  - The risk function under ranking metrics becomes smooth under this formulation, and η affect the Lipschitzness -- standard generalization bound apply : $\mathcal{O}\left(K^2/(\eta\sqrt{n})\right)$
  - The error due to the approximation is: $\mathcal{O}\left(\exp\left(-\min_{i\neq j}(f(x_i)-f(x_j))^2/\eta\right)\right)$

  tradeoff for smoothness and generalization

- The unexamined ***ranking consistency***
  - Being consistent in the risk-minimization sense ≠ giving optimal ranking under infinite data!
  - NP-hard for ranking consistency? Another tradeoff ... [DMJ10]

Domain challenges using IR feedback data

- Perhaps the biggest domain challenge of using IR feedback data for learning is the ***missingness / sparsity*** of feedback
  - Limited slots to show recommendation or respond to queries
  - If we view the feedback as ***supervised signals*** (other views will be discussed later), then handling the missingness is critical
  - If we use feedback for ***data exploratory / analysis tasks***, sparsity will be a major blocker
- Revisiting some of the classical ideas in IR
  - ***Transductive SVM***: low-density separation
  - ***Collaborative filtering***: clustered user interest
  - ***Latent semantic indexing / matrix factorization***: distributional hypothesis (entity with similar distribution should have similar representation), shared low-rank hidden structure
  - ***PageRank / TF-IDF***: numerical statistics on the importance of an entity
  - …

Handling unlabeledness

- *Transduction* and *semi-supervised* learning
  - Both aims to achieve good performance on $(x_{n+1}, \ldots, x_{n+m})$ based on $\{(x_i, y_i)\}_{i=1}^n$
  - Supervised learning aim at: $x_{n+1} \sim P_X$, P_X unknown
  - Does knowing the testing samples in advance make a difference?

    -- if P_X, $P_{Y|X}$ are related, and $\{x_i\}_{i=1}^{n+m}$ provides a better estimation of P_X than $\{x_i\}_{i=1}^n$
  - Semi-supervised learning focus on finding good decision rule $f : \mathcal{X} \to \{-1, 1\}$

    Transductive learning focus on finding a set of good predictions: $\{\hat{y}_i\}_{i=n+1}^{n+m}$
- Provable guarantee? Hardly any without assumptions ...
  - Just say what if X is cause of Y, and P_X is generated independently from $P_{Y|X}$
- What realistic assumptions lead to the success of some classical IR methods?
  - P_X has a ***smooth density***, and decision boundary lies in the ***low-density region***
  - The optimal decision boundary on $P_{X_{\text{train}}}$ and $P_{X_{\text{test}}}$ does not shift much, so ***pseudo-labelling*** may work well

Handling unlabeledness

- The distributional or data geometry assumption by semi-supervised learning all imply some degree of similarity between $\{x_i\}_{i=1}^{n+m}$ and $\{x_i\}_{i=1}^{n}$
  - What if they are naturally dissimilar due to the selection process?

    -- samples with certain property are less likely to be labelled

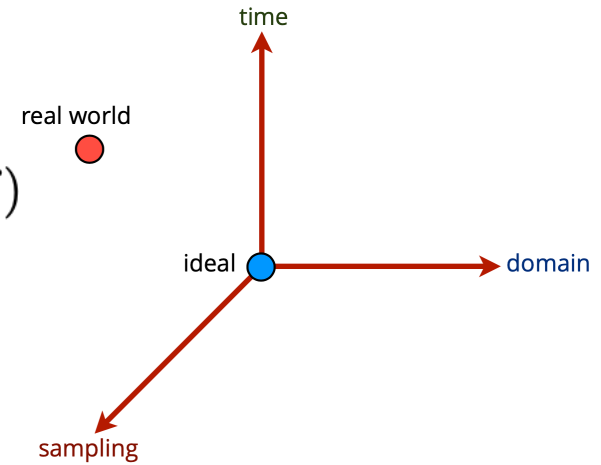
    -- the testing sample we are interested in are under the influence of new system

    -- ...
- Domain adaptation: train $\hat{f}_P$ on *source domain* P, and achieving small risk $R_Q(\hat{f}_P)$ on target domain Q
  - The generalization error bound usually depends on two terms:

    (a). *some (hypothesis-driven) domain discrepancy:* $D_{\mathcal{F}}(P, Q)$

    (b). *the smallest joint error that can be achieved*: $\min_{f \in \mathcal{F}} R_P(f) + R_Q(f)$
  - Can add *domain-discrepancy regularization* to make (a) small,

    but (b) will depend on the overlapping, i.e. $\text{supp}(Q) \subset \text{supp}(P)$ [JSR19]



Handling sparsity*

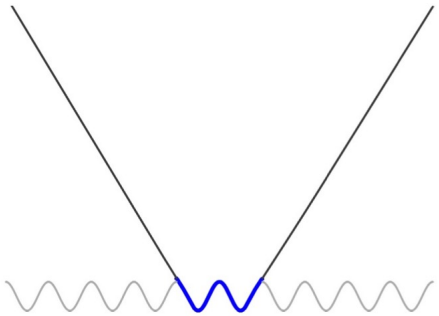
- Intuition behind using ***representation learning*** or ***graph neural network*** (GNN) to address *sparsity*:
  - The collected feedback themselves reveal some *structural similarity* among samples
  - The loss function + model structure of representation learning and GNN *implicitly characterizes those similarity* (e.g. pointwise mutual information, Weisfeiler-Lehman Distance, shortest-path metrics)
  - When over-parameterized (e.g. by layering up just like standard NN) and optimized by GD, we may again enter the *implicit regularization* regime
  - The prediction of the models are ***interpolation results***
  - The learnt representation correspond to the mapping of the ***feature-learning kernel***

*: The background and set up are somewhat tedious, so we defer the detailed discussions to our online material

Debias, extrapolation

- Relative feedback are less prone to bias than point-wise feedback, but:
  - Immune to all of *selection bias, exposure bias, popularity bias, position bias*, ... ?
  - Can possibly address all the bias in learnt patterns?
  - Does it provide a scientific way to quantify the effect of bias?
  - What about cases not represented in the collected data?

- Beyond the *support* of collected data:
  - We studied so far i.i.d generalization, which is intrinsically *interpolation*
  - *"... It takes all sorts to make a world ..."* -- interpolation doesn't satisfy all
  - Do not wish overparameterized models to *extrapolate* smartly – they behave like *linear functions* outside the support (recall from *NTK theory*).



[XZL+20]

- It's time to consider how to unleash the power of *intervention:*
  - A way to exchange plausible assumption for plausible conclusion.

Summary of Part 1

- Understanding what makes overparameterized NN work:
  - **Expressivity**: Generating smoother decision boundaries that *interpolate the data*
  - **Optimization**: Allowing GD to go beyond interpolation point -> bringing *implicit regularization*
  - **Generalization**: implicit regularization + interpolation -> optimum solution with *smaller capacity*
- Making NN work ≠ good performance for IR tasks
  - *Memorization* of NN -> less *fairness* for *under-represented group*
  - *Importance weighting* may not achieve the expected effect
- Some extra complications for ranking problems
  - What we learnt in classification mostly carry to ranking problems
  - For effective optimization, must suffer the tradeoff between *smoother loss* and *bias*
  - Consistency of risk minimization ≠ consistency of ranking
- IR domain challenges deserve more attention
  - Modern ML solutions often suffer from them, not magically solving them
  - The merits of introducing domain knowledge to solve the impossible

Part 2 – Intervention and Causal Inference

- Understanding the effect of an intervention can address many hard problems in IR system
  - *Consequence* of changing algorithm, data pipeline, webpage design, ...
  - *Knowledge* about how users make decision (mechanism of the environment)
  - *Long-term utility / fairness* of our decision

- Standard statistical models no longer satisfy this purpose, because:
  - Intervention can be hypothetical and violating the natural course of observed data
  - Intervention can create alternative interpretations that may or may not be captured by regular rules, e.g. by conditional probability.

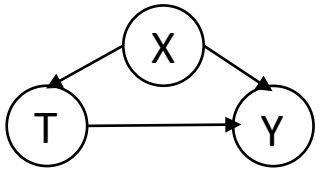
- The language of *causal inference* fills in the gap
  - Significantly emphasizes intervention within existing probability framework.

Pearl & Rubin causal model

- Recall that we wish to characterize everything related to *making intervention*. The solution from *Pearl's structural causal model*:
  - *Joint distribution* of the data, generated from the basic *noise variable* $\{U_i\}_{i=1}^{d}$
  - A *collection of equations* that formalize the *assumptions* of how the variables interact, e.g.

$$X_i := f_i(Pa(X_i), U_i), i = 1, \ldots, d$$

  - A *graphical model* that represent the assignment structure



  - *Assigning* values to certain variables specify a *response function,* via *do-operation*

$$p(Y \mid do(T := t)) \quad (\text{different from } p(Y \mid T = t))$$

  - *Average causal effect* of an intervention -> difference in the substitutions to the assignment:

$$\mathbb{E}[Y \mid do(T := 1)] - \mathbb{E}[Y \mid do(T := 0)]$$

Pearl & Rubin causal model

- From *conditional statement* to *interventional statement*
  - The biggest disagreement occurs with ***confounding*** – doing X=x may change something else and fail to coincide with conditional probability
  - But we can *control for* the confounding factors (marginalization):

$$p(Y = y \,|\, do(T = t)) = \sum_{x \in \mathcal{X}} p(Y = y \,|\, T = t, Pa(T) = x) \cdot p(Pa(T) = x)$$

  - The above *adjustment formula* allows us to estimate average causal effect from data. What about causal (***counterfactual***) questions other than the causal effect?
    - Observed evidence -> propagate the evidence to update the posterior of exogenous variables, e.g. $p(Pa(T) = x) \rightarrow p'(Pa(T) = x)$
    - Perform do-operation as usual with the updated distributions

Pearl & Rubin causal model

- If the assignment is *randomized* and the *intervention* takes the form of (binary) *treatment*, Rubin's model focus on the *potential outcome*
  - With n *units*, $(Y_1(i), Y_0(i)),\ i = 1, \ldots, n$ give the outcome under treatment/o.w.
  - It reflects the effect of intervention (treatment) more directly --

$$T(i) \in \{0, 1\} \text{ as boolean treatment indicator, then } Y(i) = T(i)Y_1(i) + (1 - T(i))Y_0(i)$$

  - *Average causal effect* can be straightforwardly estimated, although we can only observe one potential outcome – suppose coin-toss assignment:

$$\mathbb{E}[Y(i) \,|\, T(i) = 1] = Y_1(i), \quad \mathbb{E}[Y(i) \,|\, T(i) = 0] = Y_0(i)$$

  - Let $\bar{Y}_1, \bar{Y}_0$ be the population average of $(Y_1(i), Y_0(i)),\ i = 1, \ldots, n$
    then *average causal effect* = $\bar{Y}_1 - \bar{Y}_0$, because

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} Y(i), \,|\, T(i) = t\right] = \bar{Y}_t \quad t \in \{0, 1\}$$

The causal inference languages

- Despite the many conceptual debates, both models are useful for IR sys:
  - What variables can be intervened?
  - Is it possible to observe all confounding variables?
  - Reliability and usefulness of the structures among variables?
  - Source of randomness?
  - ...
  - A/B testing, offline studies, explainable IR, bias / fairness, ...

- *Causality and intervention* beyond average causal effect:
  - Which direction – $p(X,Y) = p(Y \mid X)p(X) \quad or \quad p(X,Y) = p(X \mid Y)p(Y)$
  - Ohm's law, altitude & temperature
  - *Intervention -> Invariance -> Independence & Causality*
  - This view will be useful for pattern recognition (later)

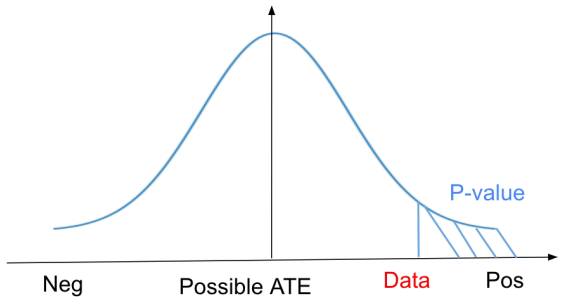
Design and inference

- How do we test whether an *intervention* can achieve desired outcome?
  - To exclude bias from all potential confounding, we design coin-toss assignment (as in Rubin's model and compute ***average causal effect***
  - If it is positive, is it positive just by chance? (*inference*: draw conclusion under uncertainty)
  - We shall use a *stochastic proof by contradiction*:

$$H_0: \text{ non-positive vs. } H_1: \text{ positive. How about } p(\text{more positive than what the data tells?} \mid H_0 \text{ is true})?$$ *

  - Intuitively, we can access $p(\text{some function of data} \mid H_0 \text{ is true})$ -- because of the design.

- *Hypothesis testing* and p-value
  - The above example is an instance of hypothesis testing
  - Central to the inference is ***p-value***



*: we use this type off non-rigorous notation for the sake of space.

Design and inference

- *Hypothesis testing, p-value, significance level, and confidence region*:
  - What criteria to use? Reject H_0 if p-val < α – then p(false positive) < α!
  - Significance level α
  - Taking a detailed look at p-val < α for *average treatment effect* $Z = \bar{Y}_1 - \bar{Y}_0$

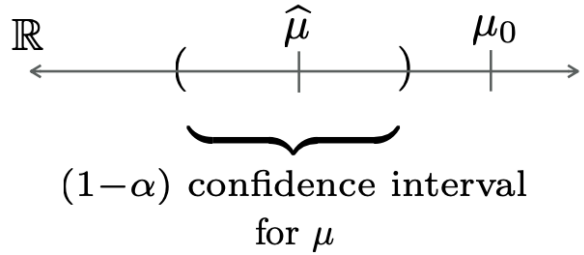
    Suppose the potential outcome are 1-subgaussian, equal sample in the *treatment group* and *control group. U*sing previous concentration result:

$$p\left(Z \le \sqrt{\frac{2\log(1/\alpha)}{n}}\right) \le \alpha \leftrightarrow \text{ reject } H_0 \text{ when } 0 \notin \left[\hat{z} - \frac{2\log(1/\alpha)}{n}, \infty\right]$$

  - There is an equivalence between rejecting based on *p-value* and *confidence region* for $H_0 : \mu = \mu_0$

$$\text{p-value} \le \alpha \leftrightarrow \mu_0 \notin [LCB(n_1, n_2, \alpha), UCB(n_1, n_2, \alpha)]$$

    where LCB, UCB are the lower/upper confidence bound.



A/B testing, metric, continuous monitoring

- When comparing two systems online, users are randomly bucketed and assigned to experience each system (A/B testing). Practically:
  - If there is a performance difference, we hope to detect it / reject the null hypothesis asap.
  - P-value is constantly checked to monitor the progress.

- The *sensitivity* of metric
  - For IR sys, online testing metrics have more room to explore
  - Reduce the variance of a single metric, or combine multiple metrics smartly
  - ***Rao-Blackwell Theorem***: using sufficient statistics to construct metric with smaller variance

$$\mathbb{E}[(\theta - \mathbb{E}[\theta \mid T])^2] \leq \mathbb{E}[(\theta - \hat{\theta})^2], \text{ for all } \hat{\theta}$$

  - ***Linear Discriminant Analysis***: linear combination of metrics that optimizes Z-score

$$\max_{\theta} \frac{\bar{Z}_1 - \bar{Z}_0}{\sqrt{var(\bar{Z}_1 - \bar{Z}_0)}} \quad s.t. \quad Z = \theta^T [Y^{(1)}, \ldots, Y^{(d)}]$$

A/B testing, metric, continuous monitoring

- Recall that *significance level* α is designed for *one-time control* of false positive rate *under fixed sample size*
  - Let $P^{(n)}$ be the p-value obtained from the *first-n* samples
  - Under null hypothesis, given a fixed n, it holds $p(P^{(n)} \leq \alpha) \leq \alpha$ (***uniformity***)
  - In *continuous monitoring*, the test is continued if p-val > α, so the real *stopping time is*

$$\tau := \min\{n \in \mathbb{N} : P^{(n)} \leq \alpha\}$$

  - Note that $p(P^{\tau} \leq \alpha)$ can be much bigger than α. (why?) False positive becomes very likely!
- How to make sure p-value is *always valid*, e.g. satisfy uniformity?
  - Recall that $\text{p-value} \leq \alpha \leftrightarrow \mu_0 \notin [LCB(n_1, n_2, \alpha), UCB(n_1, n_2, \alpha)]$
  - The previous confidence regions are derived for the average of i.i.d variables
  - Under continuous monitoring, $Y_1, \ldots, Y_n$ are dependent, so $\bar{Y}$ is a random walk.
  - Using concentration bound for random walk!

Continuous monitoring, best-arm identification

- Develop confidence regions for random walks -> *always valid* p-value:
  - Confidence band for i.i.d average: $\mathcal{O}(\sqrt{\frac{log(1/\alpha)}{n}})$
  - Confidence band for random walk average: $\mathcal{O}(\sqrt{\frac{\log\log n + C\log(1/\alpha)}{n}})$ [ZZS+16]
  - Achieves $p(\bigcup_{n\in\mathbb{N}} \mu_0 \notin [LCB(n,\alpha), UCB(n,\alpha)]) \leq \alpha$ under null hypothesis $H_0 : \mu = \mu_0$
- By making p-value any-time, we have more choices with adaptive testing:
  - *Adaptively* update the pool of candidates, without sacrificing the rigor of testing
  - Be smarter in traffic directing – good candidates deserves more samples, while exploring bad candidates just enough to safely eliminate / replace them
  - ***Best-arm identification*** algorithm that also relies on upper/lower confidence bounds

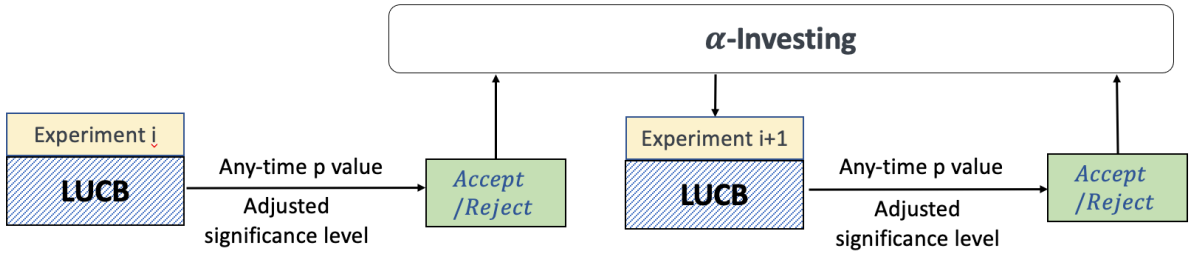
Best-arm identification, sequential testing

- Best-arm identification with pure exploration methods:
  - Always-valid p-value allows us be more adaptive (creative) in finding the best candidate system
  - ***LUCB*** method for pure exploration
    1. obtain equal amount of initial feedback for each system
    2. keep the traffic to the *current-best*, *second-best* and the *control system* (a^*, a^{**}, a^0)
    3. Update a^*, a^{**}, and iterate with step 2 until: [*LCB of* a^*] > [*UCB of* a^{**} *and* a^0]

- Integrate testing with LUCB
  - Can we just compute the always-valid p-value when LUCB stops, and decide how to proceed?
  - If we do this multiple times, the significance level α no longer guarantees the *online false discovery rate (FDR)* of the sequence of tests. (why?)
  - The significance level also needs to be updated every time an *accept/reject decision* is made
  - ***α*-investing** method for online FDR control: "invest" (discount) α each time when a testing is called, "reward" (increase) α when making discovery. [FS08]

Best-arm identification, sequential testing

- What cause the inherent difficulty of the algorithm? It has been shown with a information-theoretical lower bound that:
  - How “spread-out” the gaps are
  - The gap between the best and second-best arm

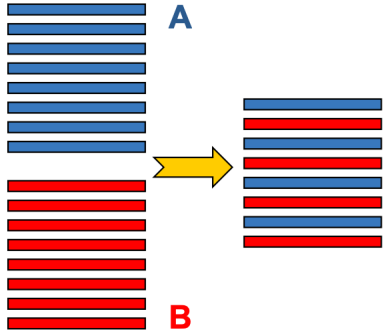


  - power? adding context? ....
- Some fundamental issues for A/B testing
  - absolute feedback, venerable to *between-subject* variability (by how much?)
  - when comparing ranking outcomes in particular, design *within-subject* relative comparison?

Interleaving and dueling

- Interleaving -- eliminate noise by letting user compare both alternatives
  - more robust to users' decision bias
  - less affect user experience
  - clicks directly reflect users' preference for A vs. B

[YJ09]

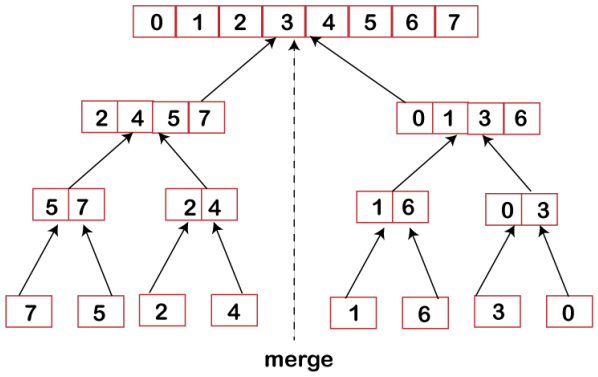


- Set up interleaved outputs – again, room to optimize *sensitivity*
  - *Balanced interleaving, Team Draft, Probabilistic Interleave*
  - A ***probability distribution*** to show a particular combination (think about randomized treatment assignment)
  - A ***scoring rule*** to interpret click -- a measure for treatment effect, $H_0 : \text{score}_A = \text{score}_B$

- *Optimization* via random user model and max-entropy principle:
  - ***Optimization*** *variable*: probability to show each page
  - ***Constraint***: a model of random user, express no preference
  - ***Objective***: maximizing the *entropy* (uncertainty of having a winner)

Interleaving and dueling

- Making interleaving test *adaptive* with K ranking systems
  - "Dueling" – create a schedule for *pairwise comparison* to find the best candidate
  - The idea of *using lower/upper confidence bounds* to compare under uncertainty is still valid
  - **Key challenge**: pairwise comparison to determine total order with K systems, $\mathcal{O}(K^2)$?
  - *Example: interleaved filter*
    - *randomly pick a^* to compare with all others*
    - *repeat until finding a^{**} whose LCB/UCB goes beyond those of a^**
    - *elimination, repeat*
- Integrating with algorithms from ***sorting***
  - Reduce the number of dueling needed to determine the order
  - Achieves $\mathcal{O}(K^2) \to \mathcal{O}(K)$
  - How many samples are needed? (topics for later)
  - Compatible with adaptive online testing? (reduction to cardinal bandit [AKJ14])



Thinking about the assumptions

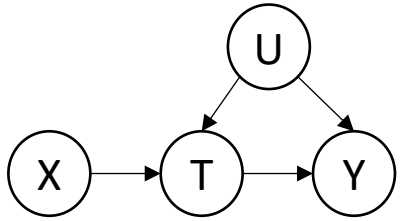
- Acute audience may find a critical background missing – what makes *randomized intervention* work for causal inference in the first place?
  - ***Stable Unit Treatment Value Assumption (SUTVA)***

    treatment one unit receives does not change the effect for another unit
  - ***Consistency***

    true outcome agrees with the potential outcome given the treatment indicator
  - ***Ignorability***

    potential outcome conditionally independent of treatment given defounding variables

- Unfortunately, they are all violated to a degree in IR sys...
  - Spillover, network and equilibrium effect
  - Leap between *exposure as measured* and *exposure as intervened*?
  - In the presence of unobserved confounding, are potential outcome ***missing-at-random (MAR)***? More importantly, can causal inference problem be treated as a missing data problem?



Observational studies and offline learning

- When the ability to launch randomized intervention is limited, or would like to mine the logged data from experimentation
  - Does the problem reduce to *pattern recognition with feedback data*?
  - Incorporate *causal knowledge* to address the *partial observability* of potential outcome?

Unit	Treatment status T_i	Outcome under treatment $Y_i(1)$	Outcome under no treatment $Y_i(0)$	Covariates X_i
1	1	✓	?	✓
2	1	✓	?	✓
3	0	?	✓	✓
4	0	?	✓	✓

  - Problem solved if we estimate "?" →
  - Studying causal problem as missing data problem attracts huge attention in IR

- Unfortunately, these two aren't the same [PM18]
  - Domain ***overlapping***?
  - ***Identifiability*** of causal mechanism (invariant mechanism)?
  - When ***imputing*** missing data require unsupported ***extrapolation ...***

Is observational studies a missing data problem?

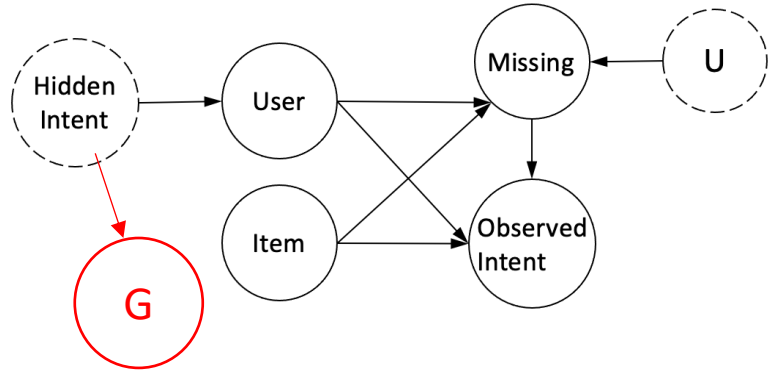
- Admittedly, if the missing mechanism is simple enough, life becomes much easier:
  - If feedback is *missing completely at random* (**MCAR**), which means the ***missing mechanism*** – which we can consider as treatment, is assigned by coin toss
  - If feedback is *missing at random* (**MAR**), the missing mechanism is still random but may depend on some confounding factors
  - In both cases, *average causal effect* can be effectively estimated using previous techniques

- When feedback is missing not at random (**MNAR**)
  - Can be caused by *missing with certainty,* e.g. some items have zero chance to be exposed
  - The crucial ***positivity*** assumption is violated, and positivity is needed to ensure ***overlap*** between treatment / control group
  - If a subgroup of subjects always receives the *same intervention*, we cannot estimate the effect of intervention changes on that subgroup *without further assumptions*
  - What further assumptions are needed? – e.g ***identifiability*** of certain causal pathways [MP21]

Is observational studies a missing data problem?

- In IR system, user selection bias -> *hidden intent* often causes the missingness of the *observable intent*
- "Self-masking" – extremely challenging
- How to use survey data to estimate the average income of low-income family when they family don't have money to install phone and answer the survey?

  -- Nobel-winning solution in Economics, *Heckman correction*



- Generally impossible to impute the missing feedback (unobserved intent) with guarantee, unless:
  - The pathway *{U -> missingness}* is known and satisfy *positivity* (e.g. fully randomized recommendations with known policy)
  - *Side information* about the hidden intent, e.g. hidden intent causes the clustering geometry of users (Node G), which is often observed in IR data
  - The pathway of *{hidden intent -> G}* , which is an *independent mechanism*, will assist extrapolation as we discuss later.

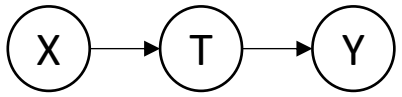
[XY22]

Double learning and targeted maximum likelihood learning

- Can we leverage modern ML methods as estimators, and plug their predictions for estimating average causal effect ψ?
  - Recall the *adjustment formula*

$$p(Y = y \,|\, do(T = t)) = \sum_{x \in \mathcal{X}} p(Y = y \,|\, T = t, Pa(T) = x) \cdot p(Pa(T) = x)$$

  - Let $X = Pa(T)$ -- use neural network to obtain $\hat{p}(Y \,|\, T, X)$
  - Big issue: most *finite-sample* ML-based estimator are biased! (e.g. the use of regularization)
    confidence interval obtained in the usual way may not cover the true average causal effect!
  - How about $\hat{\psi}^Q := 1/n \sum_{i=1}^{n} (\hat{Q}(1, x) - \hat{Q}(0, x)), \quad \hat{Q}(t, x) = \mathbb{E}[Y \,|\, t, x]$?

    -- not using 

: X affect Y only through treat assignment $g(X) := p(T = 1 | X = x)$

  - Sufficiency of ***Propensity Score*** [RR83]:

$$\psi = \mathbb{E}\big[\mathbb{E}[Y \,|\, g(X), T = 1] - \mathbb{E}[Y \,|\, g(X), T = 0]\big]$$

Double learning and targeted maximum likelihood learning

- *Semi-parametric estimation theory* to the rescue under *unknown* propensity scores
  - Recall that we estimate $\hat{Q}(x,t) = \mathbb{E}[Y \mid X = x, T = t]$, $\hat{g}(x) = p(T = 1 \mid X = x)$
  - These estimation from ML models could be *biased*, hurting the *asymptotic properties* (e.g. confidence interval may not cover the true average causal effect ψ)
  - Fortunately, $\hat{\psi}$ can still have good asymptotic property if satisfies [Ken16]:

$$\frac{1}{n}\sum_i \varphi(y_i, t_i, x_i; \hat{Q}, \hat{g}, \hat{\psi}) = 0$$

  - "*Non-parametric estimating equation*" with "*efficient influence curve*" (think of first-order bias under Taylor expansion)
  - ***Targeted maximum likelihood learning*** [VR06] – solve the estimating equation by perturbing $\hat{Q}$ using some parametric submodel, e.g. $\hat{Q}^{(1)} = \hat{Q} + \epsilon H(\hat{g})$, H is given, and use MLE to estimate ϵ
  - ***Double/debiased ML*** [CCD+18]*:* use a ***Neyman-orthogonal score*** equation for first-order debias, computing a cross-fitted augmented IPW estimator
  - As long as $\hat{Q}$ and $\hat{g}$ are faster than $n^{1/4}$, then $\hat{\psi}$ enjoys $n^{1/2}$-rate *asymptotic normality*

Propensity weighting method and counterfactual learning

- When treatments are characterized by *known* distributions, address changing the treatment on a population as switching the distribution
  - Easily verify mathematically that: $\mathbb{E}[Y \mid do(T=1)] = \mathbb{E}\big[YT \,/\, \mathbb{E}[T=1|X=x]\big]$
  - The many causal assumptions, especially positivity, ensures the *unbiasedness* of the estimation:

    $$\mathbb{E}[Y \mid do(T=1)] - \mathbb{E}[Y \mid do(T=0)] = \mathbb{E}\Big[Y\Big(\frac{T}{\pi(X)} - \frac{1-T}{1-\pi(X)}\Big)\Big],\ \pi(X) = p(T=1 \mid X=x)$$

  - What about the variance? (often need *strong positivity / overlapping*)
- If we have a good estimation for the *average potential reward* using collected data, can we use that estimation to learn a good policy?
  - ***Counterfactual learning*** : $\arg\max_{\pi_\theta} \mathbb{E}[Y \mid do(T=\pi_\theta(X))]$
  - Moving from treatment to action $a \in [K]$. Propensity score thus becomes: $\pi_0(A=a_i \mid X=x_i)$

    so we have: $\hat{R}(\pi_\theta) := \hat{\mathbb{E}}[Y \mid do(\pi_\theta(A|X))] = \frac{1}{n}\sum_i y_i \frac{\pi_\theta(a_i|x_i)}{p_i}, \quad p_i = \pi_0(a_i \mid x_i)$

Propensity weighting method and counterfactual learning

- Is counterfactual learning a supervised learning problem with weighted loss?
  - Not exactly. The observations are different: $(x_i, \text{instructive } y_i^*)$ versus $(x_i, a_i, p_i, \text{evaluative } y_i)$
  - Access to the loss are different: $\ell(y_i^*, y)$ known, versus $\ell(y, f(x_i))$ unknown for $y \neq y_i$
  - Despite the conceptual difference, what makes counterfactual learning difficult?

hypothesis + optimization

- *Distribution shift* and *variance of risk estimator* for hypothesis
  - We've seen before in LTR the *Bernstein-type bound* – variance of risk estimation matters!
  - The variance is going to be large with small probability
    on denominator ...
  - ***Clip*** small probability, ***renormalize*** the propensities, or ***penalize the variance*** in general

$$\mathcal{O}\left(\sqrt{\frac{V_n(f)\log(\mathcal{C}(\mathcal{F})/\delta)}{n}}\right)$$

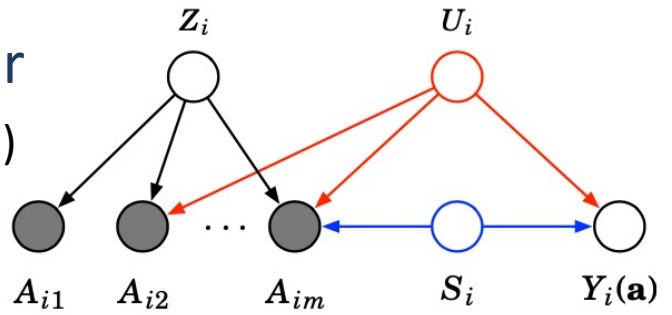
- Importance of having a *baseline* for optimization
  - If all feedback are non-positive, what will happen? (*degeneracy* of the learning problem because a upper bound can be trivially found)
  - Find a good $y_i' = y_i - r(a_i, x_i)$ to make the learning and optimization more *robust*.

[SJ15]

Multiple causes, deconfounding, robust optimization

- We talked about unobserved confounding makes inference from observational data infeasible
  - may be compatible with many potentially contradictory causal explanations
  - how much information about unobserved confounding can be recovered from observed data?

- Infer a latent variable as a substitute for unobserved confounder
  - Suppose there are multiple causes (e.g. *MF factor models* in IR [WLC+18])
  - Assume *SUTVA, overlap*, and some parametric forms
  - If we can find such a *proxy* Z, then we can safely ignore U [WB19]
  - Can employ some *encoder-decoder* learning framework (trade assumptions for assumptions)

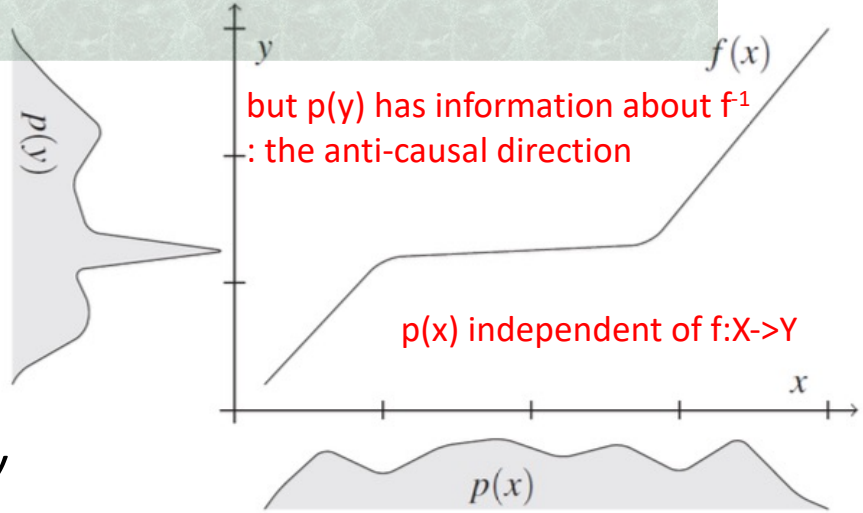


- How sensitive are the outcome?
  - Adding ad-hoc violations to the causal assumptions, and investigate the resulting perturbations
  - Or making the learning/optimization *robust to the violation* of causal assumptions [XRK+20]

Connection to IR pattern recognition

- Causality and learning (plenty in IR)
  - Predict target from its *cause* vs. from its *effect*
  - The principle of ***independent causal mechanism***

    [PJS17] -- independent mechanism + autonomous modules, and they do not inform each other
  - Exploit the *independent mechanisms*, e.g. via *causal discovery* or *directional learning*, and use them to assist generalizing to unseen data



- Uncertainty quantification, learn-then-test

$$p(Y_{\text{test}} \in S(X_{\text{test}})) \geq 1 - \alpha$$

  - Creating statistically rigorous prediction sets for ML predictions (IR cares *coverage*!)
  - ***Distribution-free conformal prediction*** – use quantiles of calibrated scores
  - Learn-then-test to optimize the *converge risk:* $\mathcal{R}(S_\lambda) := p(Y_{\text{test}} \notin S_\lambda(X_{\text{test}}))$

    suppose the *coverage set* S_λ depends on a parameter λ [AB22]
  - Hypothesis testing for whether the risk is controlled for a particularλ + *FDR control*

$$\{H_\lambda : \text{ the risk is controlled at } \lambda;\ \lambda \in \Lambda\} \implies \text{p}(\sup_{\lambda \in \Lambda} \leq \alpha) \geq 1 - \delta$$

Summary of Part 2

- Exploiting the intervenability of IR systems
  - Designed experiments -> answering causal questions
  - Problem structure + policy optimization -> gaining *efficiency* for IR experiments
- Counterfactual reasoning under domain practice
  - How valid are the *causal assumptions* under common IR practices?
  - May be more difficult than a *missing data problem*
  - Again, might need to incorporate *domain knowledge* for the rescue
- Offline pattern recognition with experimental feedback data
  - Knowledge about *intervention* makes learning from evaluative feedback favorable
  - Be aware of the variance caused by *distribution mismatch*
  - *Causality* can assist learning, but we may have to make deals with the devil of *confounding*

Part 3 – Action and Sequential Decision Making

- Problems in IR system can require modelling the *long-term consequences* of actions
  - many problems are best described as optimizing a *sequence of actions*
  - target becomes $r_t + Q(s_t, a_t)$ -- adding a long-term value component
  - We use "***action***" in the context of optimizing a system objective, and "***intervention***" in the context of causal inference – they *probe the environment* for different purposes
  - *pattern recognition* and *causal inference* techniques become means to achieve more complex goals
  - *Things need to be "learnt" in order to "act/control"*

- *Dynamic system* and *control*
  - Imagine we know how does our *action* a_t change the user's ***state***: $S_{t+1} = f(S_t, a_t, \epsilon_t)$
  - We can access to (sampling) the user's feedback (***reward***) to our action a_t: $p(R_t(S_t, a_t) = r)$
  - Can we device a sequence of actions to maximize the reward of the whole *trajectory*?

Dynamic system and planning

- *Dynamic system* and *control (*assume the user will interact with us (*horizon*) for T *rounds)*
  - Use everything collected to produce the next action, with a *policy:* $A_{t+1} = \pi_t(S_0, A_0, \ldots, S_{t+1})$
  - The optimization problem becomes :

$$\text{maximize}_\pi \quad \mathbb{E}[\sum_{t=0}^{T} R_t(S_t, A_t)] \qquad \text{subject to: } S_{t+1} = f_t(S_t, A_t, \epsilon_t),\ A_t = \pi_t(S_0, A_0, \ldots, S_t)$$

  - Can be solved by ***dynamic programming*** -- recursively find optimal policy by starting the final time and going backwards to solve for earlier stages.
  - Let the Q-function be the best achievable value from step t1 to t2:

$$Q_{t1,t2}(s,a) = \max_{a_{t1},\ldots,a_{t2}} \big[\sum_{t=t1}^{t=t2} R_t(S_t, u_t)\big], \quad (S_{t1}, a_{t1}) = (s,a)$$

  - Since the terminal Q function is $Q_{T,T}(s,a) = \mathbb{E}[R_T(s,a)]$, we recursively update:

(Bellman equation) $$Q_{t,T}(s,a) = \mathbb{E}\big[R_t(s,a) + \max_{a'} Q_{t+1,T}(S_{t+1}, u')\big], \quad S_{t+1} = f_t(s,a,\epsilon_t)$$

the best action to take, i.e. optimal policy is $\pi_{t+1}(s) = \arg\max_{a'} Q_{t+1}(s,a')$

Online learning with evaluative feedback

- Practically, we don't know how our action changes users' states $S_{t+1} = f(S_t, a_t, \epsilon_t)$, and still want to optimize the cumulative reward:
  - Suppose the reward is $R_t(a, s) = \langle \phi(a, s) - \phi^*, \theta_t \rangle$, ϕ^* is a *comparator*

    if we can observe θ_t (*instructive feedback*) – ***standard online learning***.
  - O.w., if we ignore states and focus only on (action, reward)? -- ***Multi-armed bandit***
  - If we believe only the state itself is useful, e.g. associate the reward of different actions, can we just model that? -- ***Contextual bandit***
  - We still believe state transition is crucial to our long-term success -- ***Reinforcement learning***

- Central to bandit problems is the *exploration-exploitation tradeoff:*
  - In the previous adaptive online testing, we do not consider the *cost of exploration* – everything serves for *inference*
  - When exploring sub-optimal actions carries a cost, and our goal is to maximize the *total reward*, need to balance finding good policy (exploration) and achieve best reward (exploitation)

Multi-armed bandit and exploration-exploitation

- A *K-armed bandit* is a collection of distributions $\{P_a : a \in \mathcal{A}\}, |\mathcal{A}| = K$
  - choose action A_t and observe feedback (reward) $R_{A_t} \sim P_{A_t}$
  - objective is the total reward
  - performance is measured by ***regret***: $\mathcal{R}_T(\pi) := n\mu^* - \mathbb{E}_\pi[\sum_{t=0}^{T} R_t]$
    where $\mu^* = \max_{a \in \mathcal{A}} \mu_a, \quad \mu_a = \mathbb{E}[P_a]$
  - A bound on the regret: how many samples are needed to achieve a policy with high reward (sample efficiency)

- Basic exploration methods and their regret bound
  - ***Explore then commit*** – choose each candidate certain number of times, and stick to the one with best empirical performance $\mathcal{O}(T^{2/3})$
  - ***UCB*** – choose the candidate with the highest upper confidence bound, observe reward, and update the bound $\mathcal{O}(KT \log T)$

Multi-armed bandit and exploration-exploitation

- When the state (context) is involved and $R_t = r(S_t, A_t) + \epsilon_t$
  - How to best incorporate context into the MAB framework?
  - Does it suffice to consider $\mathcal{R}_T = \mathbb{E}\big[\sum_{t=1}^{T} \max_a r(S_t, a) - \sum_{t=1}^{T} R_T\big]$? (how are context chosen?)
  - A cleaner way is to think of an imaginary feature map s.t. $r(s,a) = \langle \theta^*, \phi(s,a) \rangle$ and consider $\mathcal{A} = span(\{\phi(s,a)\}) \subset \mathbb{R}^d,\ A_t \in \mathcal{A}$
  - Becomes an instance of linear bandit $\mathcal{R}_T = \mathbb{E}\big[\sum_{t=1}^{T} \max_{a \in \mathcal{A}} \langle \theta^*, a \rangle - \sum_{t=1}^{T} R_T\big]$
  - Can similarly use UCB-type exploration (***LinUCB***) – construct a confidence set $\mathcal{C}_t$ for $\hat{\theta}$ and let $UCB(t,a) = \max_{\theta \in \mathcal{C}_t} \langle \theta, a \rangle$.
    If $\mathcal{A}$ is believed to be continuous, then consider $A_t = \arg\max_{a \in \mathcal{A}_t} UCB_t(a)$.

$\mathcal{O}(d\sqrt{T \log T})$

Multi-armed bandit and exploration-exploitation

- We just formulate the *linear bandit* problem, but
  - Is constructing the confidence set as simple as computing variance of *least-square estimator*?

    Note that actions are not fixed nor independent ...
  - *Martingale* methods will be involved (high non-trivial) ...
  - Consider each action as recommendation, context as user feature: even if we embed everything s.t. $\mathcal{A} \subset \mathbb{R}^d$, the action space is discrete in nature: $|\mathcal{A}| = K$

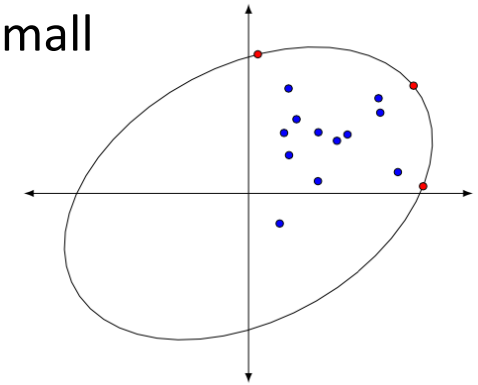
    so it is infeasible to find $A_t = \arg\max_{a \in \mathcal{A}_t} UCB_t(a)$ for exploration.

- Exploration in the Euclidean space: *D-optimal* experiment design
  - Finding exploratory actions $a_1, \ldots, a_T$ such that the confidence region for $\hat{\theta}$ is small
  - It is shown equivalent to finding a *minimum-volume ellipsoid*

    that contains all the points: $\log\det \sum_{a \in \mathcal{A}} \pi(a) a a^T \ s.t. \ \sum_{a \in \mathcal{A}} \pi(a) = 1$

[LS20]

$\mathcal{O}(\sqrt{dT \log(KT)})$ $\left(\text{versus } \mathcal{O}(d\sqrt{T \log T})\right)$



Multi-armed bandit and exploration-exploitation

- *Importance weighting* and ***Boltzmann exploration***
  - Recall using IPW estimator to unbiasedly estimate the reward of an unplayed action

$$\hat{R}_t(a) = \frac{\mathbb{1}[A_t = a]}{\pi_t(a)} r_t$$

  - Let $\hat{S}_t(a)$ be the estimated total reward for an action, and consider the *Boltzmann (softmax) exploration* policy

$$\pi_t(a) = \frac{\exp(\eta \hat{S}_{t-1}(a))}{\sum_{i=1}^{K} \exp(\eta \hat{S}_{t-1}(i))}$$

  - Converge exponentially fast toward actions with very large reward $\mathcal{O}(\sqrt{TK \log K})$
- *Follow the regularized leader* and ***mirror descent*** for linear bandit
  - Can we make the update more like online GD? – *perturb the leader* for exploration

$$a_{t+1} = \arg\min_{a \in \mathcal{A}} \left(\eta \sum_{t=1}^{T} \langle a, \hat{\theta}_t \rangle + F(a) \right), \hat{\theta} \text{ is IW-estimated}$$

$\mathcal{O}(\sqrt{dT \log T})$

  - Has a corresponding "GD" update formulation under suitable regularization function $F(a)$

$$a_{t+1} = \arg\min_{a \in \mathcal{A}} (\eta \langle a, \hat{\theta}_t \rangle + D(a, a_t))$$

The power of structure

- The structure of domain IR problems may significantly increase sample efficiency
  - Suppose the action space represents a list of m ranked items – combinatorically large is we don't exploit the structure in ranking ($\mathrm{K}' = K!\,(K-m)!$)
  - Available domain knowledge: click can depend on 1). underlying quality of the item, 2). the local of the item in the ranking
  - E.g. *cascade model*: $\mu(a,k) = \mathrm{attract}(a(k)) \cdot \mathrm{examine}(a,k)$
  - Can design policy that exploits this structure

[KSW+15]

$$\mathcal{O}(K'T\log T) \text{ v.s. } \mathcal{O}(\sqrt{m^3 KT\log T})$$

  - Suppose there is an auxiliary *feedback graph* G_t such that neighboring actions have the same reward.
  - A modification of *EXP3* can improve the regret exploiting graph structure

let $\alpha(G_t)$ be the mas-number of graph:

$$\mathcal{O}\left(\sqrt{\textstyle\sum_{t=1}^{T} \alpha(G_t)\log K}\right)$$

[Val16]

The power of structure

- Representation, kernel, and linear bandit
  - Recall "*the kernel regimes for deep learning models*"
  - Structure of (action, context) can be considered more generally as defining a *similar measure* that can induce a kernel:
    $$K\big((s,a),(s',a')\big) = \phi(s,a)^T\phi(s',a')$$
  - $\phi : \mathbb{R}^d \to \mathcal{H}$ is the feature lift associated with a RKHS
  - Think of the duality between standard regression and kernelized regression
  - Sometimes coming up with a ***good kernel*** is easier than designing a good representation

    let $\tilde{d}$ be the *effective dimension* of the RKHS, ***Kernelized UCB*** can achieve: $\mathcal{O}(\sqrt{\tilde{d}T})$

[CG17]

- The arsenal of exploration methods → exploring the unknown dynamics
  - From *regret* bound to ***PAC-error*** bound: $(\delta,\epsilon) - PAC$: $V^{\pi^*} - V^{\pi_T} \leq \epsilon$ w.p. $1-\delta$

$$\mathbb{E}[\sum R_t(S_t,\pi^*(S_t))] - \mathbb{E}[\sum R_t(S_t,\pi_t(S_t))] \text{ versus } \mathbb{E}[\sum R_t(S_t,\pi^*(S_t))] - \mathbb{E}[\sum R_t(S_t,\pi_T(S_t))]$$

Markov decision process and reinforcement learning

- MDP reframes a dynamic system using *transition probabilities* and *conditional probability*:
  - *State transition* $S_{t+1} \sim P(S \mid s_t, a_t)$, *initial state distribution* $\rho_0(S)$
  - Bring horizon to infinity, add ***discount factor*** $\gamma \in [0, 1)$ and consider ***stationary policy*** (why?),
  - A MDP is given by: $\mathcal{M} := (\mathcal{S}, \mathcal{A}, P, R, \gamma, p_0)$
  - The(state) *value function* of a policy is: $V^\pi(s) = \mathbb{E}_{a_t \sim \pi(s_t)}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 = s\right]$
  - Similarly, the (state-action-value) *Q-function* is: $Q^\pi(s, a) = \mathbb{E}_{a_t \sim \pi(s_t)}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$
- Some useful notions
  - ***Visitation measure***: $d^\pi_{p_0}(s, a)$
  - *Generative model* (ability to actively sample from P or R), *offline RL* -- i.i.d data of $\{(s, a, s', r)\}$
  - ***Advantage function*** $A^\pi(s, a) := Q^\pi(s, a) - V^\pi(s)$
  - *Trajectory* distribution: $p^\pi_\rho(\tau) = \rho(s_0)P(s_1|s_0, a_0)\pi(a_1|s_1)\ldots$

Why refinforcement learning might work?

- *Stochastic approximation* of Q-function
  - A generalization of the *strong law of large number*
  - Would like to solve $x = H(x)$, but only access $H(x_i) + \epsilon_i$
  - Consider $x_{t+1} = x_{t+1} + \alpha_t D(x_t, \epsilon_t)$, e.g. $D(x_t, \epsilon_t) \coloneqq H(x_t) + \epsilon_t - x_t$, then under standard regularity conditions, it holds:
    $$x_t \overset{a.s.}{\to} x^*$$
  - Note that: $Q(s_t, a_t) = \mathbb{E}[R(s_t, a_t) + \gamma \max_{a'} Q(S_{t+1}, u')] \approx R(s_t, a_t) + \gamma \max_{a'} Q(s_{t+1}, u')$
  - Consider the update: $Q^{new}(s_t, a_t) = (1 - \alpha) Q^{old}(s_t, a_t) + \alpha\big(R(s_t, a_t) + \gamma \max_{a'} Q^{old}(s_{t+1}, u')\big)$
  - "Model-free", directly approximate optimal control cost with DP.
  - "Off-policy", can use ϵ-greedy as *control policy,* but may update based on *non-played action*
    -- $r_t + Q^{old}(s_t, a_t)$ v.s. $r_t + \max_{a'} Q^{old}(s_t, a')$
  - How is the collected information used?
    -- only use 1 piece of information per update ...

Why refinforcement learning might work?

- ***Certainty equivalence*** for discrete MDP
  - No model? Apply some exploration strategy to estimate the transitions: $\hat{P}(s' \mid s, a)$
  - Use the estimated dynamics for planning – iterative update according to *Bellman equation*

$$Q_{t+1}(s,a) = \mathbb{E}[R(s,a) + \gamma Q_t(s', \pi_k(s))], \text{ where } \pi_t(s) = \arg\max_{a'} Q_t(s,a'),\ s' \sim \hat{P}(s' \mid s, \pi_t(s))$$

  - *"Model-based", "On-policy"*
  - Denote this solution by $\pi_{\hat{P}}$, then the error is bounded as

$$V^*(s) - V^{\pi_{\hat{P}}}(s) \leq \frac{2\gamma}{(1-\gamma)^2} \sup_{s,a} \left| \mathbb{E}_{\hat{P}}[V*] - \mathbb{E}_P[V*] \right|$$

  - Note that it essentially becomes the error between P and $\hat{P}$
  - By using such as *UCB*, we can again using concentration bounds to obtain further sample complexity bound. With $N = |\mathcal{S}||\mathcal{A}|n$, n is the sample for estimating each transition triplet

price for horizon $\mathcal{O}\left(\frac{2\gamma}{(1-\gamma)^3} \sqrt{\frac{|\mathcal{S}||\mathcal{A}| \log(2|\mathcal{S}||\mathcal{A}|/\delta)}{N}} \right)$

The hateful horizon and distribution shift

- Consider we wish to learn from *N collected trajectories* $\tau = (s_0, a_0, r_0, \ldots, s_{T-1}, a_{T-1}, r_{T-1})$ offline, suppose the generating policy was exploration enough
  - Assume a uniform-exploring policy! Again, apply IW estimator for off-policy estimation

$$\hat{V}^{\pi} = |\mathcal{A}|^T \mathbb{E}_{\tau \sim unif}\left[\mathbb{1}(\pi(s_0) = a_0, \ldots, \pi(s_{T-1}) = a_{T-1}) \sum_{t=0}^{T-1} r(s_t, a_t)\right]^*$$

  - Can use the (uniform policy)-collected trajectories to obtain $\hat{\pi} := \arg\max_{\pi \in \Pi} \hat{V}^{\pi}$
  - The *PAC bound* have exponential dependency on horizon: $\max_{\pi \in \Pi} V^{\pi} \leq V^{\hat{\pi}} + T|\mathcal{A}|^T \sqrt{\frac{2}{N} \log \frac{|\Pi|}{\delta}}$
- What about lower bound, or parameterize the value functions as linear?
  - Upper bound can be loose, but lower bound also shows we need $N = \Omega(|\mathcal{A}|^H)$ to achieve

$$\max_{\pi \in \Pi} V^{\pi} \leq V^{\hat{\pi}} + 0.5 \text{ w.p. at least } 1/2$$

[WAS21]

  - Consider ***linear realizable*** cases

$$\begin{cases} Q_t^{\pi}(s,a) = (\theta_t^{\pi})^T \phi(s,a) & \Omega((d/2)^T) \\ Q_t^{*}(s,a) = (\theta_t^{*})^T \phi(s,a) & \Omega(\min\{2^d, 2^T\}) \end{cases}$$

Exponential! Why?
Horizon
Distribution shift

*: we ignore the initial state distribution for simplicity

The hateful horizon and distribution shift

- What *structure/distribution assumptions* are needed to get rid of the exponential dependency on horizon?
  - If it holds: 1). ***Linear Bellman Completeness***

$$\exists w \text{ s.t. } w^T \phi(s,a) = r(s,a) + \mathbb{E}_{s' \sim P_t(s,a)} \max_{a'} \theta^T \phi(s',a')$$

2). ***D-optimal exploration***

$$\arg\max_\pi \log\det \mathbb{E}_\pi[\phi(s,a), \phi(s,a)^T]$$

  - Consider using *least-squares value iteration*

for $t = T-1, \ldots, 0$:

$$\theta_t = \arg\min_\theta \sum_{(s,a,r,s') \in D_h} \left(\theta^T \phi(s,a) - r - V_{t+1}(s')\right)^2 \quad \text{and set } V_t(s) = \max_{a \in \mathcal{A}} \theta_t^T \phi(s,a)$$

  - The resulting policy $\hat{\pi}_t(s) = \arg\max_a \theta_t^T \phi(s,a)$ returns $(\delta, \epsilon) - PAC$ with sample complexity:

$$\Omega\left(T\left(d^2 + \frac{H^6 d^2 \log(1/\delta)}{\epsilon}\right)\right)$$

[ZLK+20]

Supervised / Representation learning challenged

- Modern learning methods are known to achieve good approximation errors, but we just saw that even exact recovery might not be good enough
  - Suppose achieving low-error recovery for the value function [DKW+20] $\begin{cases} |Q_t^*(s,a) - \theta_t^T \phi(s,a)| \le \delta \\ |Q_t^\pi(s,a) - (\theta_t^\pi)^T \phi(s,a)| \le \delta \end{cases}$
  - When $\delta = \Omega(\sqrt{T/d})$, algorithms that return $(0.9, 1/2) - PAC$ needs $\Omega(\min\{|\mathcal{S}|, \exp(d\delta^2/16)\})$
- What if we use model-based learning instead of value-based, assuming approximate linear MDP?
  - The transitions and reward satisfy: $|p(s' \mid s,a) - \theta(s')^T \phi(s,a)| \le \delta, \quad |\mathbb{E}[R_t(s,a) - \theta_t^T \phi(s,a)]| \le \delta$
  - Unfortunately, same story as above... $\Omega(\min\{|\mathcal{S}|, \exp(d\delta^2/16)\})$
- How about assuming good approximation on the policy itself: $\pi^*(s) \in \arg\max_a \langle \theta_t, \phi(s,a) \rangle$
  - How do we optimize the policy? (spoil alert: $\Omega(\min\{2^d, 2^T\})$ without further assumptions)

Directly policy optimization

- Directly parameterize the policy π_θ, what is the gradient for $V^{\pi_\theta}(\rho) = \mathbb{E}_{\tau \sim p_\rho^{\pi_\theta}}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$
  - Recall that $p_\rho^{\pi_\theta}(\tau) = \rho(s_0) P(s_1|s_0, a_0) \pi_\theta(a_1|s_1) \dots$
  - *Advantage function* expression: $\nabla V^{\pi_\theta}(\rho) = \frac{1}{1-\gamma} \mathbb{E}_{s \sim d^{\pi_\theta}} \mathbb{E}_{a \sim \pi_\theta} \left[A^{\pi\theta}(s, a) \nabla \log \pi_\theta(a|s) \right]$
  - *Non-concave* for both direct and softmax parameterization, but we've seen before that smoothness will give *convergence to stationary points*
  - When the gradient is estimated by samples, similar to SGD, the convergence rate is controlled by $\mathcal{O}(\sqrt{\sigma^2/T})$
  - The gradient variance σ can be very large! *Rao-Blackwellization* may help a bit ...

- Assume we can access exact gradient – when does policy gradient work?
  - Under the softmax parameterization, *gradients can easily vanish at suboptimal parameters* due to *lack of exploration* (why?)
  - Exponential scaling...

$$\frac{\partial V^{\pi_\theta}(\rho)}{\partial \theta_{s,a}} = \frac{1}{1-\gamma} d_\rho^{\pi_\theta}(s) \pi_\theta(a|s) A^{\pi_\theta}(s, a)$$

Directly policy optimization

- How about increasing initial exploration, or avoid gradient vanishing?
  - Indeed, if we optimize under *another distribution* $\kappa(s) > 0, \forall s$, policy gradient will eventually converge for softmax parameterization for $V^{\pi_\theta}(\kappa)$.
  - Restricting the growth of the *absolute values of parameters* as they tend to infinity is also useful
  - Adding a ***log-barrier regularization*** to the original objective:

$$\mathrm{KL}(Unif, \pi_\theta(\cdot \mid s)) = \frac{1}{|\mathcal{S}||\mathcal{A}|} \sum_{s,a} \log \pi_\theta(a|s) + \log |\mathcal{A}|$$

  - It holds that :

$$V^*(\rho) \leq V^{\pi_\theta}(\rho) + \mathcal{O}\left(\frac{1}{1-\gamma} \left\| \frac{d_\rho^{\pi^*}}{\kappa} \right\|_2\right)$$

distributioin mismatch
importance of the initial distribution

[AKL+21]

  - There are other families of methods (natural policy gradient) we haven't mentioned
  - But, is policy gradient actually a "gradient" method?

Gradient and random search -- zeroth-order method

- Policy gradient comes with great generality, and we just discussed how and why it works by assuming such as *bounded variance*, *direct access*, and some others...
  - Is there a tradeoff for its generality? Recall that:

$$\nabla V^{\pi_\theta} \propto \sum_{t=0}^{T-1} \left[A^{\pi_\theta}(s_t, a_t) \nabla \log \pi_\theta(a_t|s_t) \right]$$

  - Don't even depend on the *dynamics* $p(s'|s,a)$.
  - There is another jump: $\max_\tau \mathcal{R}(\tau)$ to $\max_{p(\tau)} \mathbb{E}_p[\mathcal{R}(\tau)]$ -- from optimizing all trajectories to all distributions. And we further parameterize it via some $p(\tau;\theta)$!
  - Will most likely optimize a *lower bound* of the objective that we really care about...

- Essence of policy gradient is to find gradient g such that $\mathbb{E}[g] \approx \nabla V(\theta)$, $V(\theta) := V^{\pi_\theta}$
  - Recall the *1-d gradient approximation* $\lim_{\delta \to 0} (V(\theta+\delta) - V(\theta-\delta))/2\delta$
  - Naive idea: let g be $\{V(\theta+\delta)/\delta, V(\theta-\delta)/\delta\}$ with equal probability, then: $\mathbb{E}[g(\theta)] \approx \nabla V(\theta)$
  - Smooth ***sphere sampling estimator*** : $\mathbb{E}_{u \in \mathbb{S}}[V(\theta+\delta u)u] = \delta/n \nabla \tilde{V}(\theta)$
  - So there are simpler alternatives*: Evolution strategy, bandit convex optimization, etc* [NS17]

Model-based vs. model-free

- A naive characterization for comparing the update of dynamics and values
  - *Update Q-function*: only one equation is updated per experience

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma \max_{a'} Q(s_{t+1}, a')$$

  - *Update dynamics*: d equations are updated per experience

$$\min_t \sum \|s_{t+1} - f(s_t, a_t)\|_2, \ \mathcal{S} \subset \mathbb{R}^d$$

- For ***linear quadratic regulator*** : $R_t(s_t, a_t) = s_t^T Q s_t + s_t^T K a_t + a_t^T C a_t, \ s_{t+1} = A s_t + B a_t + \epsilon_t$
  - Look like a collaborative-filtering system to you?
  - A $\mathcal{O}(d)$ gap in sample complexity for model-based and model-free RL. [TR18]

- ***Approximability by NN*** in 1-d continuous state space, for Q-function, policy, and dynamics
  - The *optimal Q-function and policy* can require *exponentially more parameters* to approximate than the dynamics. [DLY+20]

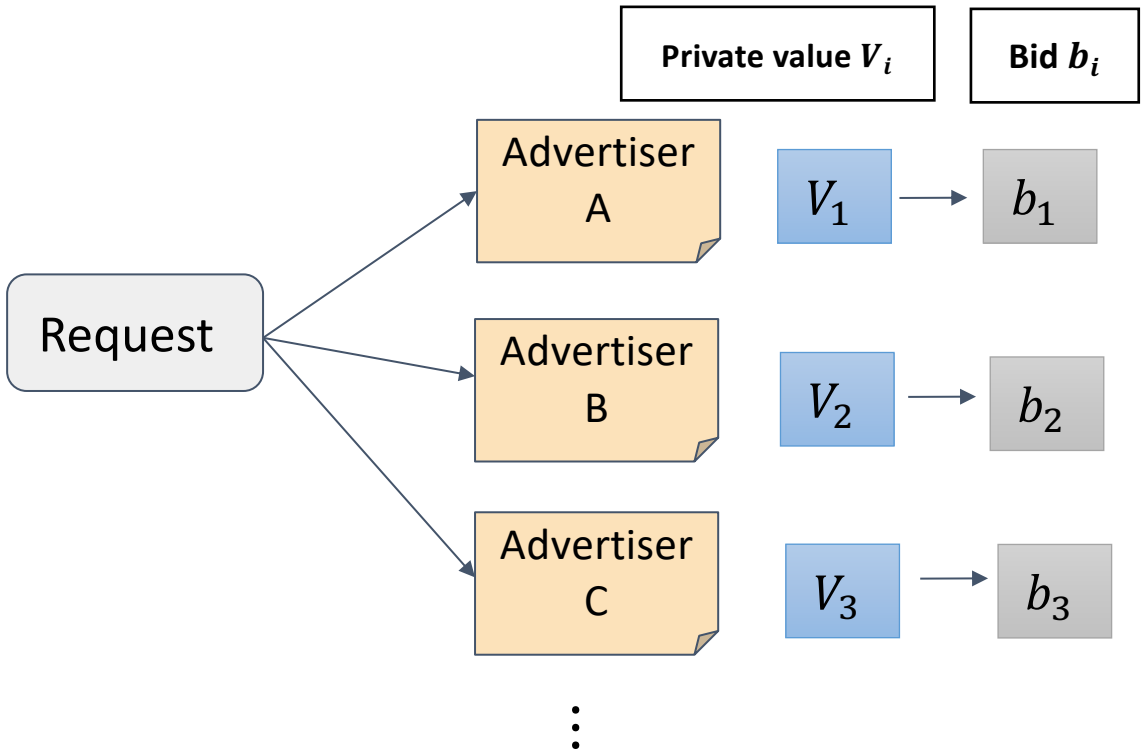
Summary of Part 3

- Viewing IR system as a *dynamic system* for decision making
  - Choosing the right setup for the right problem (*environment* and *state* are non-trivial for IR)
  - *Exploration* in IR are often more complicated – exploiting *problem structures* for efficiency
- Reinforcement learning will just work? It depends ...
  - Certainty equivalence + exploration -> success in *online learning*
  - Horizon + distribution shift -> exponential sample complexity for offline learning
  - Need strong modelling and distributional assumptions (satisfiable in IR?)
- Traditional ML wisdom can be misleading
  - Good approximation from supervised / representation learning ≠ strong RL performance
  - Gradient methods still work, but *policy gradient* is more like random search
  - More capable policy / value function classes may not make up for the absence of a *environment model* (but again, what is that for IR problems?)

Part 4 – System Design

- Interaction between system components
  - Example – auction system with second-order pricing (presented by *Bo Yang*)
  - **Two-stage** *Exploration and offline learning*?
- Using pre-trained embedding (model)
  - The hidden debt of *negative sampling*
  - Mismatch between *model architecture*
  - Calibrating model output
- Some topics on decentralized system

Interaction between system components
-- Auction Example (Generalized second-price)



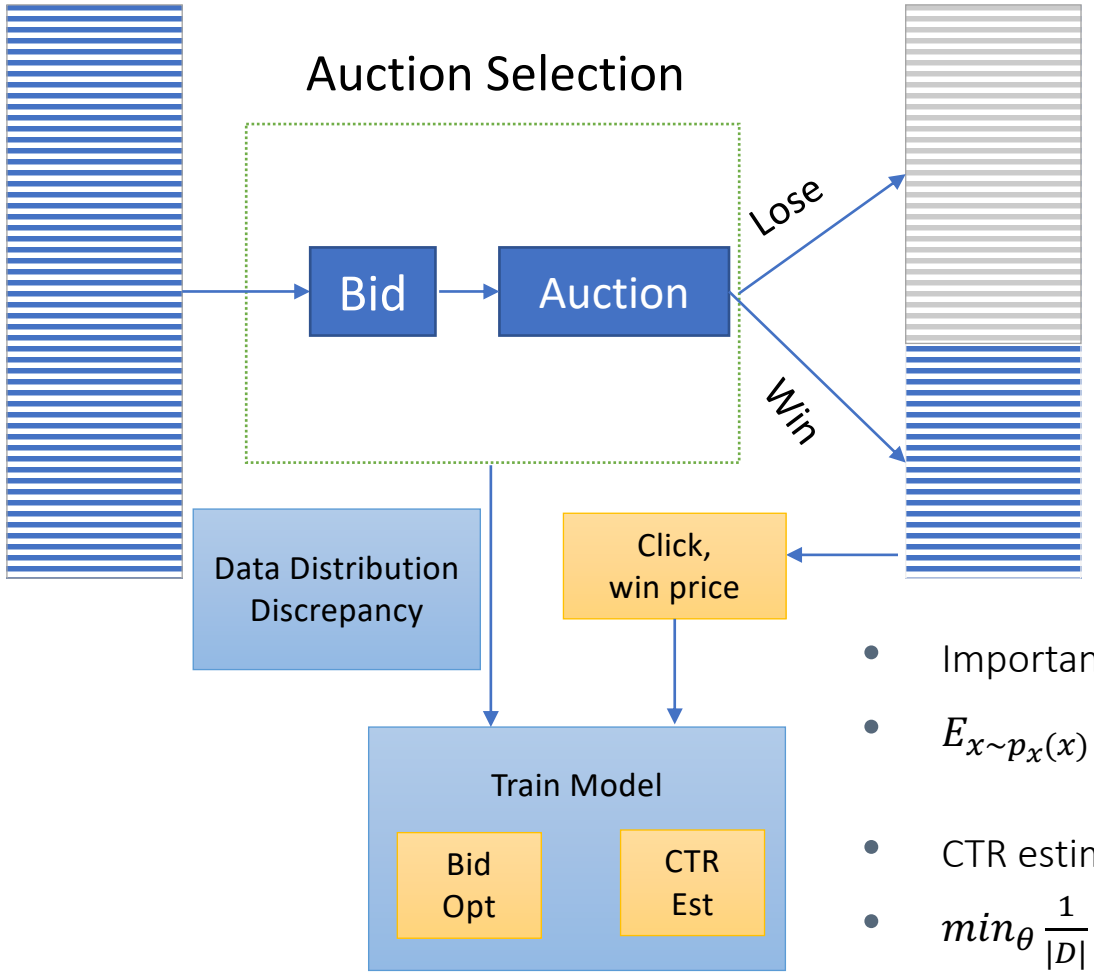
- K ad slots (K unknown in advance)
  - Top K bidders shown ads in feed
  - Each bidder pays the second highest bid
  - Truthful bidding is a dominant strategy
  - Expected reward is maximized when $b_i = V_i$

Ranking

- Advertisers are ranked according to bid and pCTR via ecpi.
- ecpi = pCTR * bid
  - pCTR = predicted click through rate (prediction model)
  - bid = price to pay if impression is clicked
  - ecpi = expected cost per impression

$$P_1(v_1, b_i, b()) = \begin{cases} v_1 - b_1 & if\, b_1 > b_i > \max\{b(v_2), \dots, b(v_{i-1}), b(v_{i+1}), \dots, b(v_n)\} \\ 0 & if\, b_1 < \max\{b(v_2), \dots, b(v_n)\} \end{cases}$$

User Response Prediction



$$q_x(x) = P(win|x, b_x) * p_x(x)$$

$q_x(x)$: Winning impression

$P(win|x, b_x)$: Prob of winning the auction

$p_x(x)$: Bid request

- The training data for user response estimation is biased and contains missing data.
  - When bid wins the auction, user response y, market price z, is observed
- Unbiased supervised learning problem
  - Loss-minimization problem on prediction data distribution $p_x(x)$
  - $\min_\theta E_{x\sim p(x)}[L(y, f_\theta(x))] + \lambda\Phi(\theta)$

- Importance sampling to reduce the bias of training data:
- $E_{x\sim p_x(x)}[L(y, f_\theta(x))] = \int_x p_x(x)L(y, f_\theta(x))dx = \frac{1}{|D|}\sum_{(x,y,x)\in D} \frac{L(y,f_\theta(x))}{1-\prod_{b_j<b_x}\frac{n_j-d_j}{n_j}}$

- CTR estimation for logistic regression:
- $min_\theta \frac{1}{|D|}\sum_{(x,y,x)\in D} \frac{-ylogf_\theta(x)-(1-y)\log(1-f_\theta(x))}{1-\prod_{b_j<b_x}\frac{n_j-d_j}{n_j}} + \frac{\lambda}{2}||\theta||_2^2$

User Response Prediction

Linear Models

Logistic regression, Bayesian probit regression.

Build the model based on the feature independence assumption.

Non-linear Models

Factorisation machine, tree models, (deep) neural networks.

More capacity of automatically learning feature interaction patterns without designing combining features.

More computational resources.

Transfer learning

Implicitly and jointly learn user's profile on both web browsing $\prod_{(x^c,y^c)\in D^c} P(y^c|x^c;\Theta)$ and ad response behaviors $\prod_{(x^r,y^r)\in D^r} P(y^r|x^r;\Theta)$.

$$\widehat{\Theta} = \max_{\Theta} P(\Theta) \prod_{(x^c,y^c)\in D^c} P(y^c|x^c;\Theta) \prod_{(x^r,y^r)\in D^r} P(y^r|x^r;\Theta)$$

Exploration and offline learning in a two-stage system?

- Most large-scale IR systems rely on the ***two-stage retrieval-ranking*** framework
  - There are usually several retrieval systems (focusing on different sources/signals) that generating the pool of candidates
  - The retrieval systems are usually *lightweight* and *using fewer signals* to reduce latency
  - A ranking system then further exploits *more complete signals* to rank the candidates
- The distribution mismatch of retrieval and ranking
  - E.g. the ***candidate distribution*** presented to the ranker is conditioned on the retrieval systems
  - E.g. the ***feedback distribution*** presented to the retrieval system is confounded by the ranker
  - E.g. different retrieval systems may exploit different ***feature distributions*** that are emphasized differently by the ranker
- Some potential consequences

[HKJ+20]
  - Common exploration strategies may suffer a *linear regret* $\mathcal{O}(T)$ due to a lack of communications between the two stages

[HKJ21]
  - *Unknown optimality* of optimizing ranking and retrieval systems *jointly vs. separately*

Using a pre-training system

- *Noise-contrastive estimation (NCE)* is often used as a unsupervised learning technique for pre-training the embeddings
  - Use ***semantic similarity*** to create ***self-supervision signal*** for learning
  - With the input of $(x, x^+, x_1^-, \ldots, x_k^-)$, the following objective is being optimized

$$\mathbb{E}_{x,x^+,\{x_i^-\}_{i=1}^k}\left[-\log \frac{\exp(\phi(x)^T\phi(x^+))}{\sum_{i=1}^k \exp(\phi(x)^T\phi(x_i^-))}\right]$$

  - The positive sample x^+ is often constructed from data, and the negative samples are obtained from *negative sampling*
  - *Cross-entropy loss* for *(extreme) multi-label classification*
- The pre-trained embeddings are then used by downstream tasks, however:
  - Imagine the downstream task uses $\phi(x)$ for *downstream classification*
  - What if a particular (x, x_i^-) turns out belonging to the *same class* in the downstream task?

The hidden debt of negative sampling

- The ***collision-coverage tradeoff***
  - Increasing #neg samples naturally leads to a better *density estimation* of the pre-training data, since it better covers the semantic space
  - But it also increases the chance of *class collision* in downstream data
  - The *class collision* can cause an excessive bias term even for generalizing to the same class-condition distribution:

$$\mathcal{O}\left(\mathcal{R}_n(\mathcal{F}) + \sqrt{\frac{\log 1/\delta}{n}} + \text{collision bias}\right)$$

- Many advanced negative sampling methods are providing *heuristic solutions* to avoid class collision:
  - Finding ***hard negatives*** to reduce the chance that (x, x_i^-) belong to the same class
  - Using ***self-training*** (self-labelling) techniques to select appropriate negative samples
- Many other confounding factors can affect the usefulness of pre-trained embedding...

Mismatch between model architectures

- In NCE, the scores are constructed from the *inner products* between embeddings
  - The *model architecture* takes a factorization format – $\langle \phi(x_i), \phi(x_j) \rangle$
  - Downstream task may consume the embedding by a different architecture, e.g. $f_\theta(x) = \theta_1 \sigma(\theta_2^T \phi(x)))$
  - What are the potential consequences of the mismatch?

- The dual kernel view of linear regression
  - Suppose $f_\theta(x) = \theta^T \phi(x)$, then the property of the solution is decided by the kernel:

$$K_\phi(x_i, x_j) := \phi(x_i)^T \phi(x_j)$$

  - A nice correspondence with the scores that are optimized during pre-training

- But for the more complicated downstream architectures
  - There could exist parameterizations where $f_\theta(x_i),\ f_\theta(x_j)$ correspond to $\langle \phi(x_i), \phi(x_j) \rangle$ in a meaningful way (recall the ***NTK theory***)
  - In general, the mismatch can cause *non-negligible bias* even when generalizing to the same class-conditional distribution

[XYK+22]

Calibrating model output

- We have seen that even if the loss function has a *probabilistic interpretation* (e.g. logistic loss), the *exponential-tail property* can lead to margin-maximization behaviors
  - The output of the pre-trained models are *margins* rather than *conditional probabilities*
  - The predicted scores thus do not reflect the ***uncertainty*** of the prediction, and will tend to ***polarize***
  - Would like $\hat{f}_\theta$ to be calibrated wrt. some reference distribution
  - If the goal is to reconstruct the uncertainty of a true probability distribution:

$$\text{find } \hat{P} := c(\hat{f}_\theta)(X) \text{ s.t. } p(\hat{Y} = y \,|\, \hat{P} = p) = p,\ \forall p \in [0, 1]$$

[GPS+17]

  - If the goal is to reconstruct the empirical user interest $\hat{Q}$:

$$\text{force } D(\hat{P} \,\|\, \hat{Q}) \text{ to be small}$$

[Ste18]

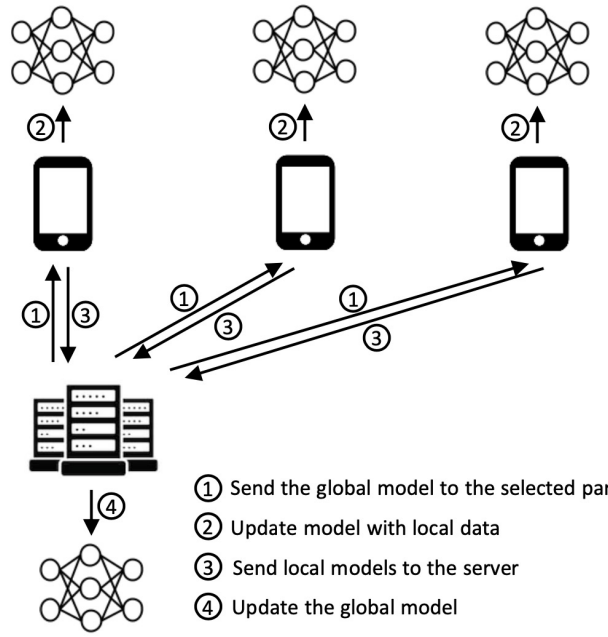
  - If the goal is to encourage invariance across environments $\{e_i\}_{i=1}^k$:

$$\text{constraint } p(\hat{Y} = y|\hat{P} = p, E = e_i) = p(\hat{Y} = y|\hat{P} = p, E = e_j),\ \forall i, j$$

[WFG+21]

Decentralized system

- *Collaborative training* of IR models across individual devices (clients) under *privacy restrictions*
  - We primarily discuss the framework of ***federated learning***
  - The key considerations aren't that different from *centralized learning*:
    - How to characterize the ***data distributions***?
    - What is the appropriate ***inductive bias*** for modelling?
    - Why does the various ***tradeoffs*** exist?
- About the source and target distributions
  - The source distribution is a *mixture* of client distribution with *known mixture probability* (e.g. given by the sample size), but it is *unknown* how the target distribution will be mixed



[MMR+17]

Decentralized system

- Following what we learnt in *Part 1*:
  - Using a large ***global model*** may cause poor performance on *under-represented clients*
  - The *unknown discrepancy* between the source and target distributions exacerbates this effect (imagine we care about generalizing uniformly across all clients)
  - *Importance weighting* may not resolve this challenge
  - A reasonable inductive bias is to employ ***personalized local models*** to complement the global model
  - But we are also subject to **privacy constraint** – even the local models are not supposed to memorize the clients' sampled data
  - Lacking ability to memorize -> less stable -> poorer overall generalization performance
- Some emerging tradeoffs for decentralized systems:
  - The traditional *bias-variance* tradeoff is now partly caused by the *relative strength* of **local vs. global** model

    [BWD+22]
  - The **privacy vs. accuracy** tradeoff now also affects the bias-variance tradeoff

Summary of Part 4

- System design has a high complexity and requires fundamental understanding of the problem and algorithm
  - *How does ML solution fit the business need*? (the auction example)
  - *What is a good inductive bias for the data distributions involved*? (the decentralized system example)
  - *How do the system components interact*? (the retrieval-ranking example)
  - *What is the gap between ML results and the desired goal?* (the calibration example)
  - ……
- The external constraints lead to a hierarchy of tradeoffs
  - The modelling and learning for pattern recognition is often the cornerstone
  - External constraints, e.g. service time, privacy, how ML outputs is used, can create several layers of algorithmic tradeoffs
- Good solutions often navigate through the problem, algorithm, and tradeoff

Some epilogue

- Embrace “universal model” vs. diving into domain knowledge
- System vs. $\sum functions$
- Mind (theory) vs. hand (engineering)

- Establish IR as a leading discipline in ML/AI
- Oceans of research topics, converting to $$
- Leading business in Web 2.0, even more so in Web 3.0?

Thank you for attending!

Da Xu
ML Manager, Staff ML Eng
Search & Recommendation
Walmart Labs

- Acknowledgement -- would like to thank:
  - Bo Yang (LinkedIn)
  - Chuanwei Ruan (Instacart)
  - Evren Korpeoglu, Sushant Kumar, Kannan Achan (Walmart Labs)
  - Many other peers, collaborators, annonymous reviewers ...
- DM for collab (research & job opportunity)
- Appreciate any feedback, comments, suggestions!

Contact: daxu5180@gmail.com (daxu5180.github.io)
Website: theoreticalfoundation4irsystem.github.io/Tutorial-KDD22

Discussions and Q&A

Part 1: Pattern Recognition

- *ML Basics*:
- *Understanding Deep Learning*
- *From classification to ranking*
- *Domain challenges of IR*

Part 2: Intervention and Causal Inference

- *The causal language*
- *Design and inference*
- *Observational studies and offline learning*
- *Connection to pattern recognition*

Part 3: Action and sequential decision making

- *Online learning with evaluate feedback*
- *The power of structure*:
- *Policy learning*

Part 4: System design

- *Interaction between system components*
- *Pre-training, negative sampling, calibration*
- *Decentralized system*

Contact: daxu5180@gmail.com (daxu5180.github.io)
Website: theoreticalfoundation4irsystem.github.io/Tutorial-KDD22

References

- [MP21] Mohan K, Pearl J. Graphical models for processing missing data[J]. Journal of the American Statistical Association, 2021, 116(534): 1023-1037.
- [AKJ14] Ailon, Nir, Zohar Karnin, and Thorsten Joachims. "Reducing dueling bandits to cardinal bandits." *International Conference on Machine Learning*. PMLR, 2014.
- [RR83] Rosenbaum P R, Rubin D B. The central role of the propensity score in observational studies for causal effects[J]. Biometrika, 1983, 70(1): 41-55.
- [PM18] Pearl J, Mackenzie D. The book of why: the new science of cause and effect[M]. Basic books, 2018.
- [Ken16] Kennedy E H. Semiparametric theory and empirical processes in causal inference[M]//Statistical causal inferences and their applications in public health research. Springer, Cham, 2016: 141-167.
- [VR06] Van Der Laan M J, Rubin D. Targeted maximum likelihood learning[J]. The international journal of biostatistics, 2006, 2(1).
- [CCD+18] Chernozhukov V, Chetverikov D, Demirer M, et al. Double/debiased machine learning for treatment and structural parameters[J]. 2018.
- [YJ09] Yue Y, Joachims T. Interactively optimizing information retrieval systems as a dueling bandits problem[C]//Proceedings of the 26th Annual International Conference on Machine Learning. 2009: 1201-1208.
- [WB19] Wang Y, Blei D M. The blessings of multiple causes[J]. Journal of the American Statistical Association, 2019, 114(528): 1574-1596.
- [WLC+18] Wang Y, Liang D, Charlin L, et al. The deconfounded recommender: A causal inference approach to recommendation[J]. arXiv preprint arXiv:1808.06581, 2018.
- [XRK+20] Xu D, Ruan C, Korpeoglu E, et al. Adversarial counterfactual learning and evaluation for recommender system[J]. Advances in Neural Information Processing Systems, 2020, 33: 13515-13526.

References

- [PJS17] Peters J, Janzing D, Schölkopf B. Elements of causal inference: foundations and learning algorithms[M]. The MIT Press, 2017.
- [AB21] Angelopoulos A N, Bates S. A gentle introduction to conformal prediction and distribution-free uncertainty quantification[J]. arXiv preprint arXiv:2107.07511, 2021.
- [XY22] Xu D, Yang B. Rethinking learning with missing feedback for recommendation. Openreview, 2022.
- [TR19] Tu S, Recht B. The gap between model-based and model-free methods on the linear quadratic regulator: An asymptotic viewpoint[C]//Conference on Learning Theory. PMLR, 2019: 3036-3083.
- [DLY+20] Dong K, Luo Y, Yu T, et al. On the expressivity of neural networks for deep reinforcement learning[C]//International Conference on Machine Learning. PMLR, 2020: 2627-2637.
- [BHM+19] Belkin M, Hsu D, Ma S, et al. Reconciling modern machine-learning practice and the classical bias–variance trade-off[J]. Proceedings of the National Academy of Sciences, 2019, 116(32): 15849-15854.
- [SHN+18] Soudry D, Hoffer E, Nacson M S, et al. The implicit bias of gradient descent on separable data[J]. The Journal of Machine Learning Research, 2018, 19(1): 2822-2878.
- [GRS18] Golowich N, Rakhlin A, Shamir O. Size-independent sample complexity of neural networks[C]//Conference On Learning Theory. PMLR, 2018: 297-299.
- [XYR21] Xu D, Ye Y, Ruan C. Understanding the role of importance weighting for deep learning. ICLR, 2021.
- [JGH18] Jacot A, Gabriel F, Hongler C. Neural tangent kernel: Convergence and generalization in neural networks[J]. Advances in neural information processing systems, 2018, 31.

References

- [SJ15] Swaminathan A, Joachims T. Counterfactual risk minimization: Learning from logged bandit feedback[C]//International Conference on Machine Learning. PMLR, 2015: 814-823.
- [HKJ21] Hron J, Krauth K, Jordan M, et al. On component interactions in two-stage recommender systems[J]. Advances in Neural Information Processing Systems, 2021, 34: 2744-2757.
- [HKJ20] Hron J, Krauth K, Jordan M I, et al. Exploration in two-stage recommender systems[J]. arXiv preprint arXiv:2009.08956, 2020.
- [XZL+20] Xu K, Zhang M, Li J, et al. How neural networks extrapolate: From feedforward to graph neural networks[J]. arXiv preprint arXiv:2009.11848, 2020.
- [JGH18] Jacot A, Gabriel F, Hongler C. Neural tangent kernel: Convergence and generalization in neural networks[J]. Advances in neural information processing systems, 2018, 31.
- [GJK21] Gatmiry K, Jegelka S, Kelner J. Optimization and Adaptive Generalization of Three layer Neural Networks[C]//International Conference on Learning Representations. 2021.
- [FZ20] Feldman V, Zhang C. What neural networks memorize and why: Discovering the long tail via influence estimation[J]. Advances in Neural Information Processing Systems, 2020, 33: 2881-2891.
- [CLV08] Clémençon S, Lugosi G, Vayatis N. Ranking and empirical minimization of U-statistics[J]. The Annals of Statistics, 2008, 36(2): 844-874.
- [RCM+05] Rudin C, Cortes C, Mohri M, et al. Margin-based ranking meets boosting in the middle[C]//International Conference on Computational Learning Theory. Springer, Berlin, Heidelberg, 2005: 63-78.
- [CW10] Chapelle O, Wu M. Gradient descent optimization of smoothed information retrieval metrics[J]. Information retrieval, 2010, 13(3): 216-235.

References

- [MMR09] Mansour Y, Mohri M, Rostamizadeh A. Domain adaptation: Learning bounds and algorithms[J]. arXiv preprint arXiv:0902.3430, 2009.
- [ZZS+16] Zhao S, Zhou E, Sabharwal A, et al. Adaptive concentration inequalities for sequential decision problems[J]. Advances in Neural Information Processing Systems, 2016, 29.
- [FS08] Foster D P, Stine R A. α-investing: A procedure for sequential control of expected false discoveries[J]. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 2008, 70(2): 429-444.
- [JSR19] Johansson F D, Sontag D, Ranganath R. Support and invertibility in domain-invariant representations[C]//The 22nd International Conference on Artificial Intelligence and Statistics. PMLR, 2019: 527-536.
- [KSW+15] Kveton B, Szepesvari C, Wen Z, et al. Cascading bandits: Learning to rank in the cascade model[C]//International conference on machine learning. PMLR, 2015: 767-776.
- [CG17] Chowdhury S R, Gopalan A. On kernelized multi-armed bandits[C]//International Conference on Machine Learning. PMLR, 2017: 844-853.
- [GPS+17] Guo C, Pleiss G, Sun Y, et al. On calibration of modern neural networks[C]//International conference on machine learning. PMLR, 2017: 1321-1330.
- [NS17] Nesterov Y, Spokoiny V. Random gradient-free minimization of convex functions[J]. Foundations of Computational Mathematics, 2017, 17(2): 527-566.
- [AKL+21] Agarwal A, Kakade S M, Lee J D, et al. On the Theory of Policy Gradient Methods: Optimality, Approximation, and Distribution Shift[J]. J. Mach. Learn. Res., 2021, 22(98): 1-76.
- [DKW+20] Du S S, Kakade S M, Wang R, et al. Is a Good Representation Sufficient for Sample Efficient Reinforcement Learning?[C]//International Conference on Learning Representations. 2020.

References

- [ZLK+20] Zanette A, Lazaric A, Kochenderfer M, et al. Learning near optimal policies with low inherent bellman error[C]//International Conference on Machine Learning. PMLR, 2020: 10978-10989.
- [LS20] Lattimore T, Szepesvári C. Bandit algorithms[M]. Cambridge University Press, 2020.
- [Ste18] Steck H. Calibrated recommendations[C]//Proceedings of the 12th ACM conference on recommender systems. 2018: 154-162.
- [WFG+21] Wald Y, Feder A, Greenfeld D, et al. On calibration and out-of-domain generalization[J]. Advances in neural information processing systems, 2021, 34: 2215-2227.
- [XY22] Xu D, Yang B. Revisiting pre-trained embedding for e-commerce machine learning. Under review, 2022.
- [WAS21] Weisz G, Amortila P, Szepesvári C. Exponential lower bounds for planning in mdps with linearly-realizable optimal action-value functions[C]//Algorithmic Learning Theory. PMLR, 2021: 1237-1264.
- [MMR17] McMahan B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data[C]//Artificial intelligence and statistics. PMLR, 2017: 1273-1282.
- [BWD22] Bietti A, Wei C Y, Dudik M, et al. Personalization Improves Privacy-Accuracy Tradeoffs in Federated Learning[C]//International Conference on Machine Learning. PMLR, 2022: 1945-1962.
- [DMJ10] Duchi J C, Mackey L W, Jordan M I. On the consistency of ranking algorithms[C]//ICML. 2010.