Mathematical Optimization Models and Applications II

Yinyu Ye
Department of Management Science and Engineering
Stanford University
Stanford, CA 94305, U.S.A.

http://www.stanford.edu/˜yyye

Chapters 1, 2.1-2, 6.1-2, 7.2, 11.3, 11.6

Unconstrained Optimization: Logistic Regression I

Similar to SVM, given the two-class discrimination training data points $\mathbf{a}_i \in R^n$, according to the logistic model, the probability that it's in a class C, say in Red, is represented by a linear/affine function with slope-vector $\mathbf{x}$ and intersect scalar x_0:

$$\frac{e^{\mathbf{a}_i^T\mathbf{x}+x_0}}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}}.$$

Thus, for some training data points, we like to determine intercept x_0 and slope vector $\mathbf{x} \in R^n$ such that

$$\frac{e^{\mathbf{a}_i^T\mathbf{x}+x_0}}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}} = \begin{cases} 1, & \text{if } \mathbf{a}_i \in C \\ 0, & \text{otherwise} \end{cases}.$$

Then the probability to give a "right classification answer" for all training data points is

$$\left(\prod_{\mathbf{a}_i \in C} \frac{e^{\mathbf{a}_i^T\mathbf{x}+x_0}}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}}\right)\left(\prod_{\mathbf{a}_i \notin C} \frac{1}{1+e^{\mathbf{a}_i^T\mathbf{x}+x_0}}\right)$$

Logistic Regression II

Therefore, we like to maximize the probability when deciding intercept x_0 and slope vector $\mathbf{x} \in R^n$

$$\left(\prod_{\mathbf{a}_i \in C} \frac{e^{\mathbf{a}_i^T \mathbf{x} + x_0}}{1 + e^{\mathbf{a}_i^T \mathbf{x} + x_0}}\right)\left(\prod_{\mathbf{a}_i \notin C} \frac{1}{1 + e^{\mathbf{a}_i^T \mathbf{x} + x_0}}\right) = \left(\prod_{\mathbf{a}_i \in C} \frac{1}{1 + e^{-\mathbf{a}_i^T \mathbf{x} - x_0}}\right)\left(\prod_{\mathbf{a}_i \notin C} \frac{1}{1 + e^{\mathbf{a}_i^T \mathbf{x} + x_0}}\right),$$

which is equivalently to maximize

$$-\left(\sum_{\mathbf{a}_i \in C} \ln(1 + e^{-\mathbf{a}_i^T \mathbf{x} - x_0})\right) - \left(\sum_{\mathbf{a}_i \notin C} \ln(1 + e^{\mathbf{a}_i^T \mathbf{x} + x_0})\right).$$

$$\boxed{\min_x \boxed{f(Ax)}}$$

Or

$$\underbrace{\min_{x_0, \mathbf{x}} \left(\sum_{\mathbf{a}_i \in C} \ln(1 + e^{-\mathbf{a}_i^T \mathbf{x} - x_0})\right) + \left(\sum_{\mathbf{a}_i \notin C} \ln(1 + e^{\mathbf{a}_i^T \mathbf{x} + x_0})\right)}.$$

This is an unconstrained optimization problem, where the objective is a convex function of decision variables: intercept x_0 and slope vector $\mathbf{x} \in R^n$.

Sparse Linear Regression Problems

We want to find a sparsest solution to fit exact data measurements, that is, to minimize the number of non-zero entries in $\mathbf{x}$ such that $A\mathbf{x} = \mathbf{b}$:

$$\begin{array}{rl} \text{minimize} & \|\mathbf{x}\|_0 = |\{j : x_j \neq 0\}| \\ \text{subject to} & A\mathbf{x} = \mathbf{b}. \end{array}$$

Sometimes this objective can be accomplished by LASSO:

$$\begin{array}{rl} \text{minimize} & \|\mathbf{x}\|_1 = \sum_{j=1}^n |x_j| \\ \text{subject to} & A\mathbf{x} = \mathbf{b}. \end{array}$$

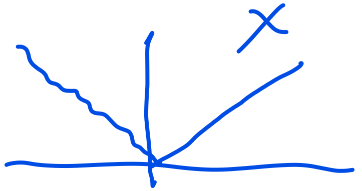
It can be equivalently represented by (?)

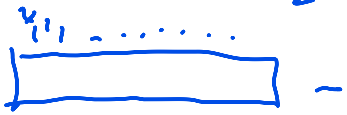
$$\begin{array}{rl} \text{minimize} & \sum_{j=1}^n y_j \\ \text{subject to} & A\mathbf{x} = \mathbf{b},\ -\mathbf{y} \leq \mathbf{x} \leq \mathbf{y}; \end{array} \quad \text{or} \quad \begin{array}{rl} \text{minimize} & \sum_{j=1}^n (x'_j + x''_j) \\ \text{subject to} & A(\mathbf{x}' - \mathbf{x}'') = \mathbf{b},\ \mathbf{x}' \geq \mathbf{0},\ \mathbf{x}\text{''} \geq \mathbf{0}. \end{array}$$

Both are linear programs!

Sparsest Data Fitting continued

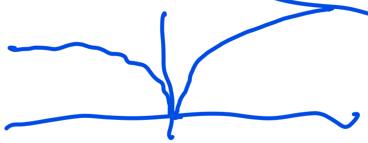
min: $\|Ax - b\|_2^2$





A better approximation of the objective can be accomplished by

$$\begin{array}{ll}\text{minimize} & \|\mathbf{x}\|_p := \left(\sum_{j=1}^n |x_j|^p\right)^{1/p} \\ \text{subject to} & A\mathbf{x} = \mathbf{b};\end{array} \quad \text{or} \quad \text{minimize} \quad \|A\mathbf{x} - \mathbf{b}\|^2 + \mu\left(\sum_{j=1}^n |x_j|^p\right)^{1/p}$$



$0 < p < 1$

for some $0 < p < 1$, where $\mu > 0$ is a regularization parameter.

Or simply

$$\begin{array}{ll}\text{minimize} & \|\mathbf{x}\|_p^p := \left(\sum_{j=1}^n |x_j|^p\right) \\ \text{subject to} & A\mathbf{x} = \mathbf{b};\end{array} \quad \text{or} \quad \text{minimize} \quad \|A\mathbf{x} - \mathbf{b}\|^2 + \beta\left(\sum_{j=1}^n |x_j|^p\right);$$

where the former is a linearly constrained (nonconvex) optimization problem and the latter is an unconstrained (nonconvex) optimization problem

Asset

Quadratic Programming (QP): Portfolio Management

For expected return vector $\mathbf{r}$ and co-variance matrix V of an investment portfolio, one management model is:

$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$

$$\begin{array}{ll} \text{minimize} & \boxed{\mathbf{x}^T V \mathbf{x}} - w\, r^T x \\ \text{subject to} & \cancel{\mathbf{r}^T \mathbf{x} \geq \mu,} \\ & \mathbf{e}^T \mathbf{x} = 1, \ \mathbf{x} \cancel{\geq} \mathbf{0}, \end{array} \quad \geq \quad \text{or simply} \quad \begin{array}{ll} \text{minimize} & \mathbf{x}^T V \mathbf{x} - \rho\, r^T x \\ \text{subject to} & \cancel{\mathbf{r}^T \mathbf{x} \geq \mu,} \\ \rightarrow & \mathbf{e}^T \mathbf{x} = 1, \end{array}$$

$x_1, \ldots \geq 0$

where $\mathbf{e}$ is the vector of all ones.

This is a (convex) quadratic program.

More CLP Examples: Robust Portfolio Management

In applications, $\mathbf{r}$ and V may be estimated under various scenarios, say $\mathbf{r}_i$ and V_i for $i = 1, ..., m$. Then, we like

$$\begin{array}{rl} \text{minimize} & \max_i \mathbf{x}^T V_i \mathbf{x} \\ \text{subject to} & \min_i \mathbf{r}_i^T \mathbf{x} \ge \mu, \\ & \mathbf{e}^T \mathbf{x} = 1, \ \mathbf{x} \ \ge \ \mathbf{0}. \end{array} \quad \Rightarrow \quad \begin{array}{rl} \text{minimize} & \alpha \\ \text{subject to} & \mathbf{r}_i^T \mathbf{x} \ge \mu, \ \forall i \\ & \sqrt{\mathbf{x}^T V_i \mathbf{x}} \le \alpha, \ \forall i \\ & \mathbf{e}^T \mathbf{x} = 1, \ \mathbf{x} \ \ge \ \mathbf{0}. \end{array}$$

This is a quadratically constrained quadratic program (QCQP). If factorize $V_i = R_i^T R_i$ and let $\mathbf{y}_i = R_i \mathbf{x}$, we can rewrite the problem as

MOSEK

$$\begin{array}{rl} \text{minimize} & \alpha \\ \text{subject to} & \mathbf{r}_i^T \mathbf{x} \ge \mu, \ \mathbf{y}_i - R_i \mathbf{x} = \mathbf{0}, \ \forall i \\ & \|\mathbf{y}_i\| \le \alpha, \ \forall i, \ \mathbf{e}^T \mathbf{x} = 1, \ \mathbf{x} \ \ge \ \mathbf{0}, \end{array}$$

which is an SOCP with additional benefits.

Portfolio Selection Problem

If no more than k stocks can be selected into your portfolio as a policy constraint?

$$\begin{array}{ll} \text{minimize} & \mathbf{x}^T V \mathbf{x} \\ \text{subject to} & \mathbf{r}^T \mathbf{x} \geq \mu, \\ & \mathbf{e}^T \mathbf{x} = 1, \\ & \mathbf{0} \leq \mathbf{x} \leq \mathbf{y},\ \mathbf{e}^T \mathbf{y} \leq \boxed{k},\ \mathbf{y} \in \{0,1\}^n \end{array}$$

$n >> k$

This is a mixed-integer quadratic program (MIP).

If the integer variables are restricted 0 or 1, it is also names as the binary optimization problem.

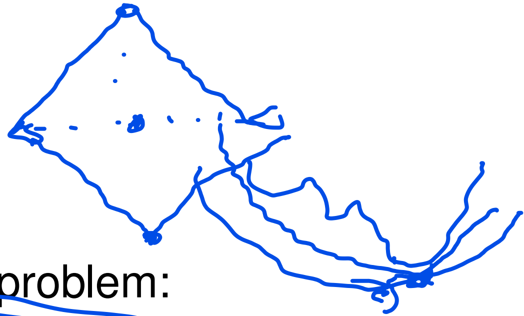
Graph Realization and Sensor Network Localization

Given a graph $G = (V, E)$ and sets of non–negative weights, say $\{d_{ij} : (i, j) \in E\}$, the goal is to compute a realization of G in the Euclidean space $\mathbf{R}^d$ for a given low dimension d, where the distance information is preserved.

More precisely: given anchors $\mathbf{a}_k \in \mathbf{R}^d$, $d_{ij} \in N_x$, and $\hat{d}_{kj} \in N_a$, find $\mathbf{x}_i \in \mathbf{R}^d$ such that

$x \in R^2$

$$\begin{cases} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 = d_{ij}^2, \; \forall\,(i,j) \in N_x, \; i < j, \\ \|\mathbf{a}_k - \mathbf{x}_j\|_2^2 = \hat{d}_{kj}^2, \; \forall\,(k,j) \in N_a. \end{cases}$$



This is a set of Quadratic Equations, which can be represented as an optimization problem:

$$\rightarrow \min_{\mathbf{x}_i \forall i} \sum_{(i,j)\in N_x} (\|\mathbf{x}_i - \mathbf{x}_j\|^2 - d_{ij}^2)^2 + \sum_{(k,j)\in N_a} (\|\mathbf{a}_k - \mathbf{x}_j\|^2 - \hat{d}_{kj}^2)^2.$$

Does the system have a localization or realization of all $\mathbf{x}_j$'s? Is the localization unique? Is there a certification for the solution to make it reliable or trustworthy? Is the system partially localizable with a certification?

Convex Relax.

It can be relaxed to SOCP (change "$=$" to "$\leq$") or SDP.

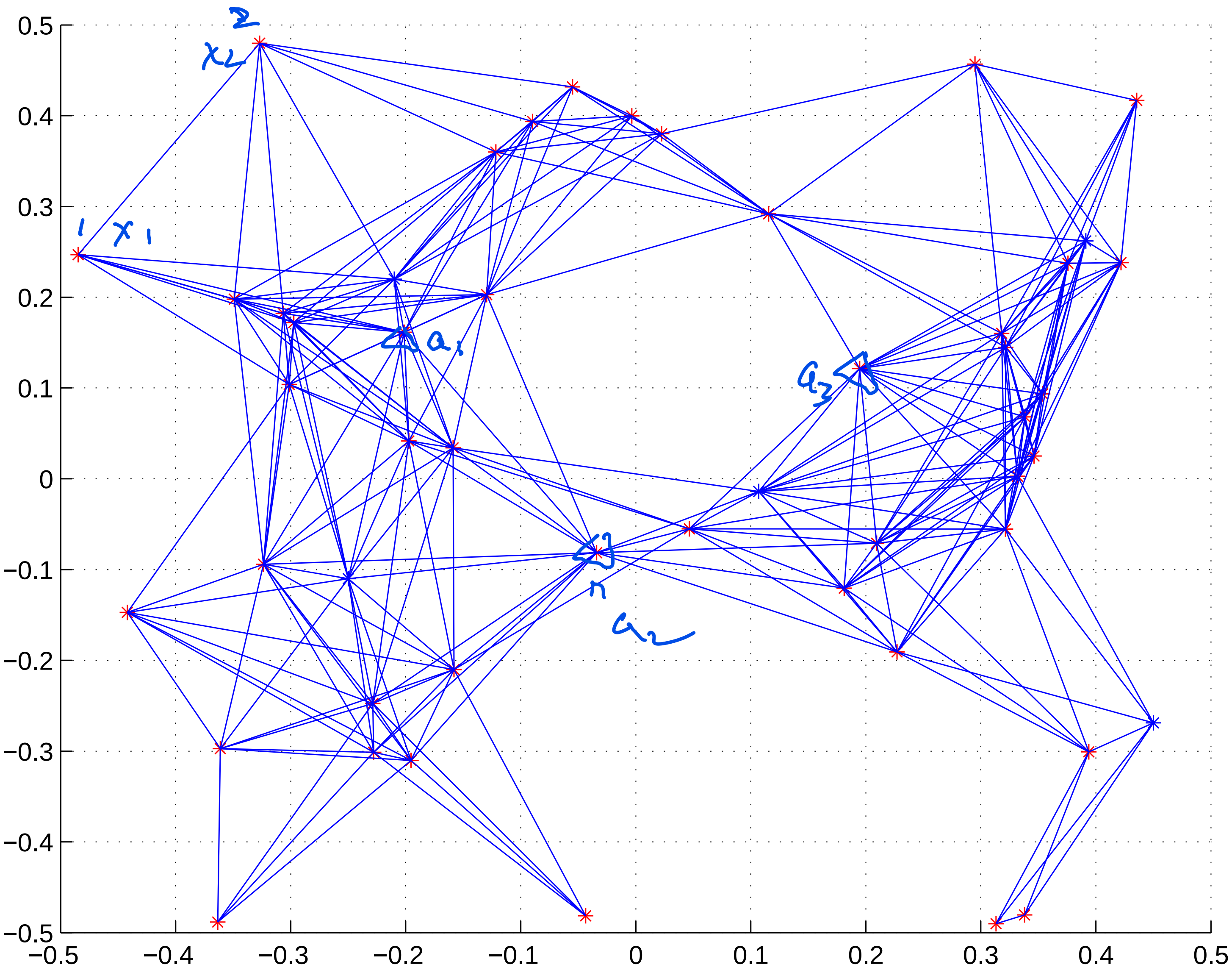


Figure 1: 50-node 2-D Sensor Localization.

Matrix Representation of SNL and SDP Relaxation

Let $X = [\mathbf{x}_1\ \mathbf{x}_2\ ...\ \mathbf{x}_n]$ be the $d \times n$ matrix that needs to be determined and $\mathbf{e}_j$ be the vector of all zero except 1 at the jth position. Then

$$\underline{\mathbf{x}_i - \mathbf{x}_j} = X(\mathbf{e}_i - \mathbf{e}_j) \quad \text{and} \quad \mathbf{a}_k - \mathbf{x}_j = [I\ \ X](\mathbf{a}_k; -\mathbf{e}_j)$$

so that

$$\rightarrow \qquad \|\mathbf{x}_i - \mathbf{x}_j\|^2 = \underline{(\mathbf{e}_i - \mathbf{e}_j)^T X^T \,|\, X(\mathbf{e}_i - \mathbf{e}_j)}$$

$$\|\mathbf{a}_k - \mathbf{x}_j\|^2 = \underline{(\mathbf{a}_k; -\mathbf{e}_j)^T [I\ \ X]^T [I\ \ X](\mathbf{a}_k;} -\mathbf{e}_j) =$$

$$(\mathbf{a}_k; -\mathbf{e}_j)^T \begin{pmatrix} I & X \\ X^T & \boxed{X^T X} \end{pmatrix} (\mathbf{a}_k; -\mathbf{e}_j).$$

$$\hookrightarrow Y$$

Or, equivalently, ~ errors

$$
\begin{aligned}
&(\mathbf{e}_i - \mathbf{e}_j)^T Y (\mathbf{e}_i - \mathbf{e}_j) = d_{ij}^2, \ \forall\, i, j \in N_x,\ i < j,\\
&(\mathbf{a}_k; -\mathbf{e}_j)^T \begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} (\mathbf{a}_k; -\mathbf{e}_j) = \hat{d}_{kj}^2, \ \forall\, k, j \in N_a,\\
&\underline{Y \succeq X^T X.}
\end{aligned}
$$

Relax $Y = X^T X$ to $Y \succeq X^T X$, which is equivalent to matrix inequality:

$\underline{Y - X^T X \succeq 0}$

$$
\begin{pmatrix} I & X \\ X^T & Y \end{pmatrix} \succeq \mathbf{0}.
$$

ran (Y) = 2

This matrix has rank at least d; if it's d, then $Y = X^T X$, and the converse is also true.

The problem is now an SDP problem: when the SDP relaxation is exact?

Algorithm: Convex relaxation first and steepest-descent-search second strategy?

Stochastic Optimization and Learning

In real world, we most often do

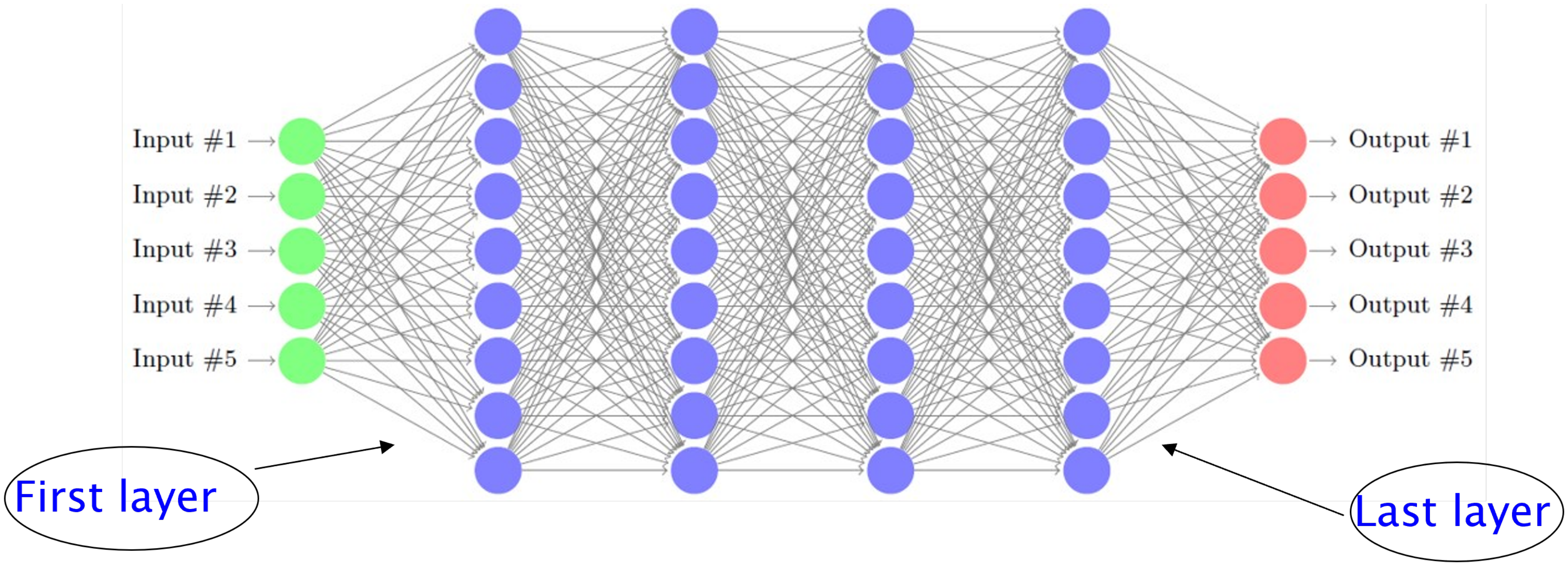
$$f(x) = E_{F_\xi} \, Loss(x, \xi)$$

$$\text{mimimize}_{\mathbf{x} \in X} \quad \mathbf{E}_{F_\xi}[h(\mathbf{x}, \xi)] \tag{1}$$

where ξ represents random variables with the joint distribution F_ξ.

- Pros: In many cases, the expected value is a good measure of performance
- Cons: One has to know the exact distribution of ξ to perform the stochastic optimization so that we most frequently use sample distribution. Then, deviant from the assumed distribution may result in sub-optimal solutions. Even know the distribution, the solution/decision is generically risky.

Deep-Learning on Neural-Network I



First layer

Last layer

The input vector is denoted by $\mathbf{x}$ and the output vector of layer l is denoted by $\mathbf{y}^l$. The edge-weights of layer l are denoted by $w^l_{i,j}$ where the relation of input-output is

$$y^l_j = \max\{0,\ w^l_{0,j} + \sum_i w^l_{i,j} y^{l-1}_j\},\ \forall j,\ l = 1, ..., L.$$

where the formula is called ReLU operator/function and $\mathbf{y}^0 = \mathbf{x}$.

Deep-Learning on Neural-Network II

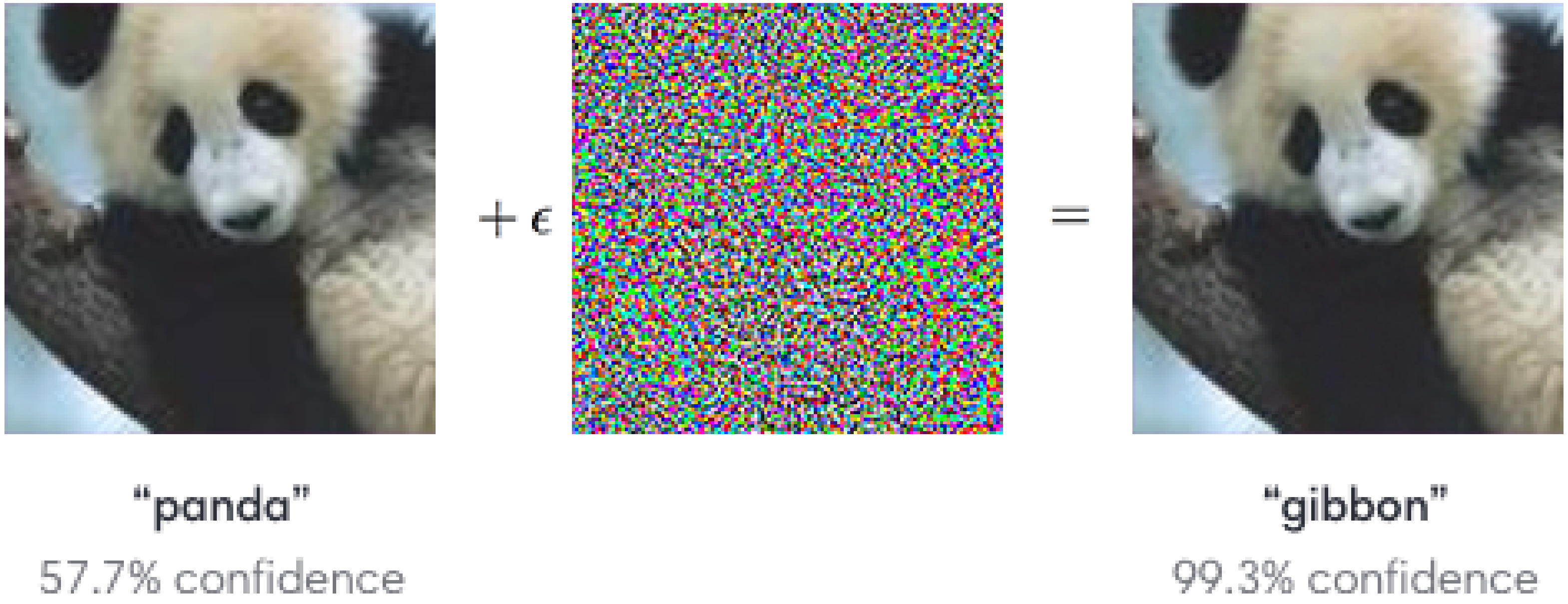
The Deep-Learning is to use massive sample images/inputs $\mathbf{x}$ to optimize/train (or learn edge-weights $w^l_{i,j}$ such that a (classification) sample-average error function is minimized. In other words, for this example, the outputs of images/inputs of Panda and Gibbon are distinguishable/separable, or they belong to different regions in the output space.

When all weights are determined, then the last-layer output vector of the neural-network, denoted by $\mathbf{y}^L(\mathbf{x})$, is a vector function/mapping of an input vector $\mathbf{x}$.

The neural network verification, for this example, is to find the smallest distortion of a given typical Panda image such that its output is in the output-region of normal Gibbon images, that is,

$$\begin{array}{ll} \text{minimize}_{\mathbf{x}} & \|\mathbf{x} - \hat{\mathbf{x}}\|^2 \\ \text{subject to} & \mathbf{y}^L(\mathbf{x}) \in \text{a (convex) region outside of } \mathbf{y}^L(\hat{\mathbf{x}}). \end{array}$$

Learning with Noises/Distortions



Goodfellow et al. [2014]

Distributionally Robust Optimization and Learning

On the other hand: Why does error occur? Believing that the sample distribution is the true distribution...

In practice, although the exact distribution of the random variables may not be known, people usually know certain observed samples or training data and other statistical information. Thus, we can consider an enlarged distribution set $\mathcal{D}$ that confidently containing the sample distribution, and do

$$\text{minimize}_{\mathbf{x}\in X} \boxed{\max_{F_\xi\in\mathcal{D}} \mathbf{E}_{F_\xi}[h(\mathbf{x},\xi)]} \tag{2}$$

In DRO, we consider a set of distributions $\mathcal{D}$ and choose one to minimize the expected value for the worst distribution in $\mathcal{D}$. When choosing $\mathcal{D}$, we need to consider the following:

- Tractability
- Practical (Statistical) Meanings
- Performance (the potential loss comparing to the benchmark cases)

This is a nonlinear Saddle-Point Min-Max optimization/zero-sum-game problem

Reinforcement Learning: Markov Decision/Game Process

$1 \cdots \longrightarrow T$

- RL/MDPs provide a mathematical framework for modeling sequential decision-making in situations where outcomes are partly random and partly under the control of a decision maker.

- Markov game processes (MGPs) provide a mathematical slidework for modeling sequential decision-making of two-person turn-based zero-sum game.

- MDGPs are useful for studying a wide range of optimization/game problems solved via dynamic programming, where it was known at least as early as the 1950s (cf. Shapley 1953, Bellman 1957).

- Modern applications include dynamic planning under uncertainty, reinforcement learning, social networking, and almost all other stochastic dynamic/sequential decision/game problems in Mathematical, Physical, Management and Social Sciences.

MDP Stationary Policy and Cost-to-Go Value

- An MDP problem is defined by a given number of states, indexed by i, where each state has a number of actions, $\mathcal{A}_i$, to take. Each action, say $j \in \mathcal{A}_i$, is associtaed with an (immeidiate) cost c_j of taking, and a probability distribution $\mathbf{p}_j$ to transfer to all possible states at the next time period.

total # s : m

total # a : n

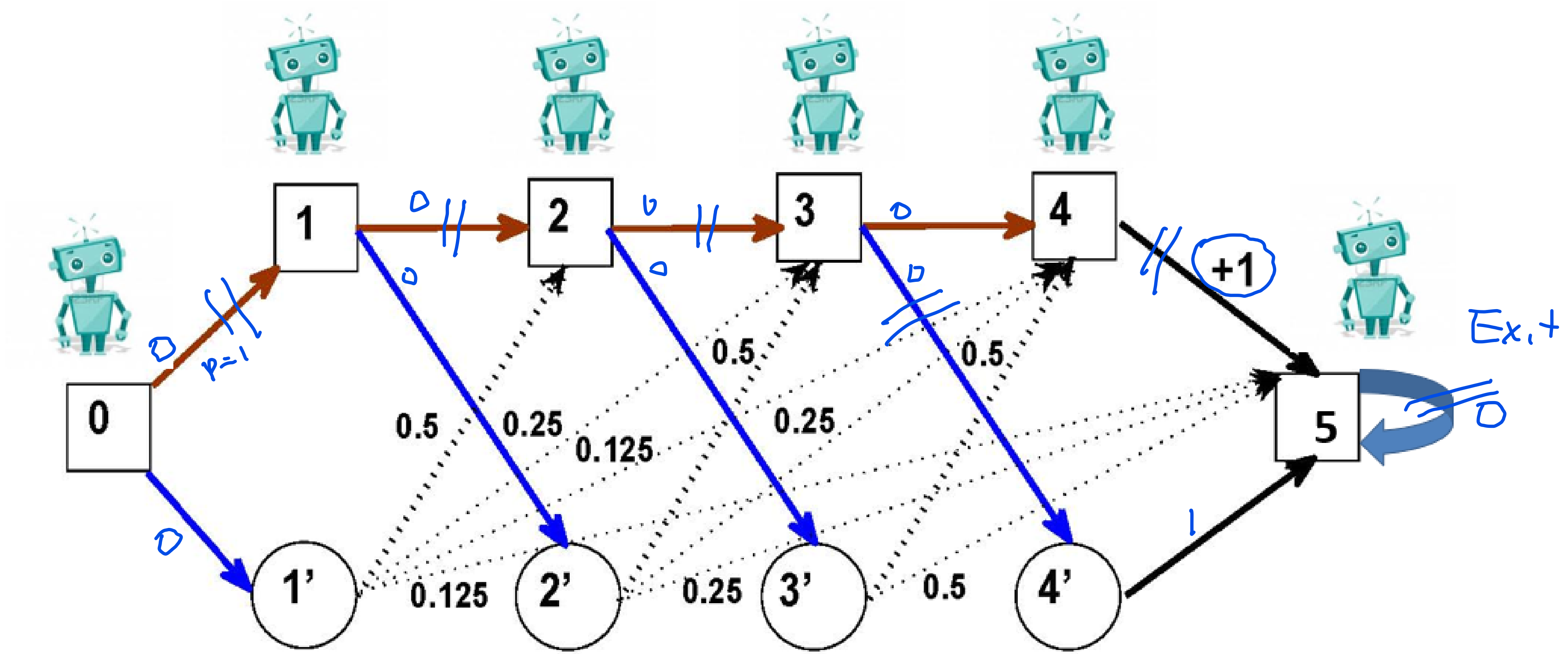
- A stationary policy for the decision maker is a function $\pi = \{\pi_1, \pi_2, \cdots, \pi_m\}$ that specifies an action in each state, $\pi_i \in \mathcal{A}_i$, that the decision maker will take at any time period; which also lead to a cost-to-go value for each state.

- The MDP is to find a stationary policy to minimize/maximize the expected discounted sum over the infinite horizon with a discount factor $0 \leq \gamma < 1$:

$$\sum_{t=0}^{\infty} \gamma^t E[c^{\pi_{i^t}}(i^t, i^{t+1})].$$

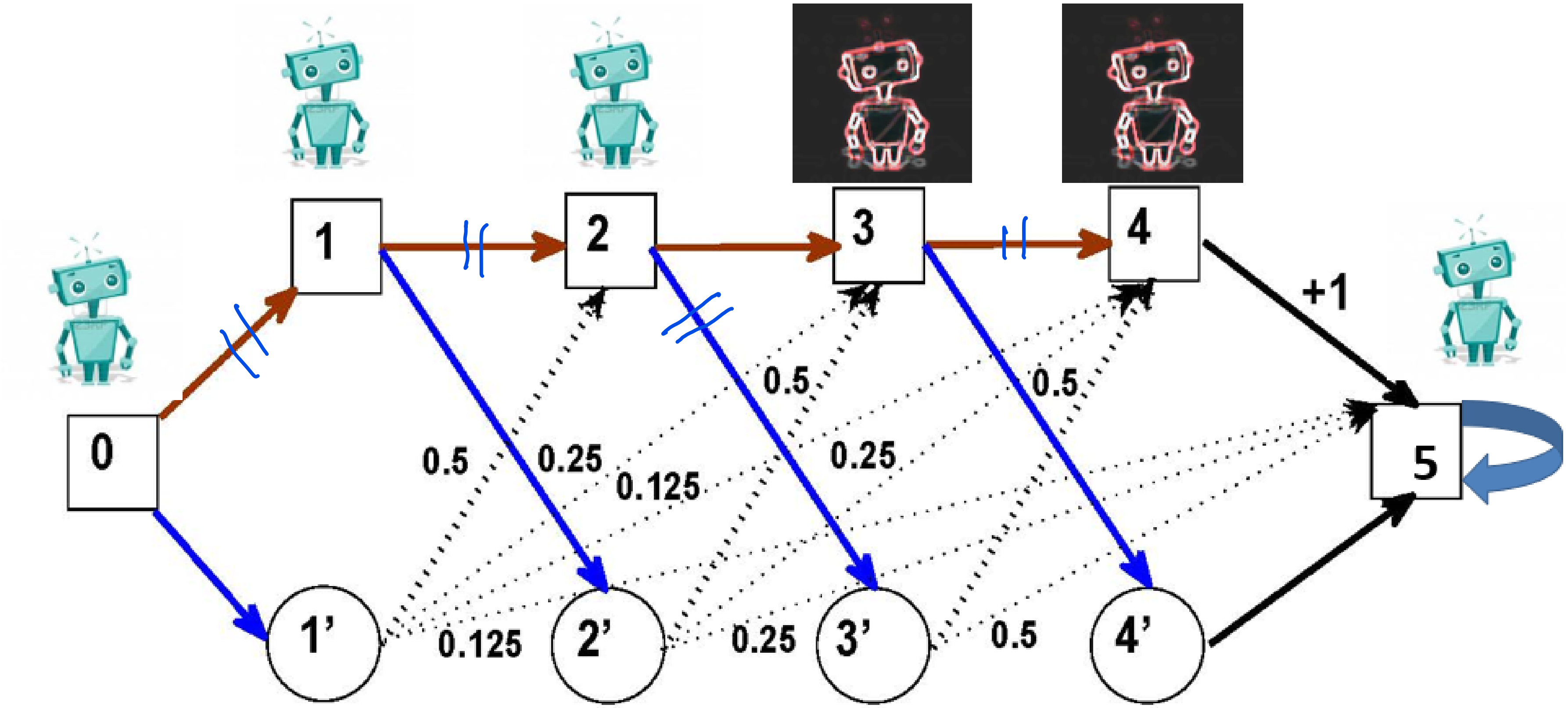
- If the states are partitioned into two sets, one is to minimize and the other is to maximize the discounted sum, then the process becomes a two-person turn-based zero-sum stochastic game.

An MDGP Toy Example: Maze Robot Runners (Simplified)



Actions are in red, blue and black; and all actions have zero cost except the state 4 to the exit/termination state 5. Which actions to take from every state to minimize the total cost (called optimal policy)?

Toy Example: Game Setting



States $\{0, 1, 2, 5\}$ minimize, while States $\{3, 4\}$ maximize.

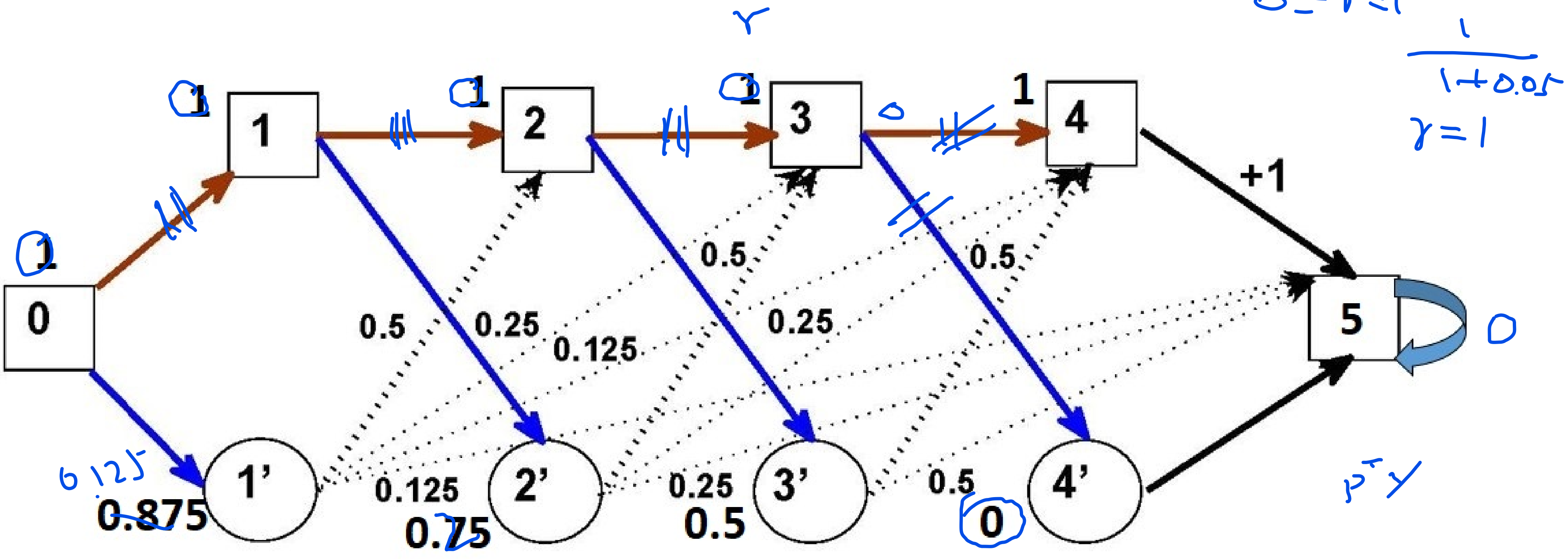
$\frac{1}{T}$ $\frac{1}{T+1}$

The Cost-to-Go Values of the States

$0 \le \gamma \le 1$

$\frac{1}{1+0.05}$

$\gamma = 1$



Cost-to-go values on each state when actions in red are taken: the current policy is not optimal since there are better actions to choose to minimize the cost.

The Optimal Cost-to-Go Value Vector

Let $\mathbf{y} \in \mathbf{R}^m$ represent the cost-to-go values of the m states, ith entry for ith state, of a given policy.

The MDP problem entails choosing an optimal policy where the corresponding cost-to-go value vector $\mathbf{y}^*$ satisfying:

max y_i

s.t

$$\boxed{y_i^* \overset{\le}{=} \underset{j\in\mathcal{A}_i}{\min}\{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^*, \ \forall j \in \mathcal{A}_i\}, \ \forall i,}$$

$$\boxed{y = T(y)}$$

with optimal policy

$$\pi_i^* = \arg\min\{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^*, \ \forall j \in \mathcal{A}_i\}, \ \forall i.$$

In the Game setting, the conditions becomes:

$\{1, \ldots, m\}$

$$\left\{\begin{array}{l} y_i^* = \underline{\min\{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^*}, \ \forall j \in \mathcal{A}_i\}, \ \forall i \in I^-, \\ \\ \text{and} \\ \\ y_i^* = \underline{\max}\{c_j + \gamma \mathbf{p}_j^T \mathbf{y}^*, \ \forall j \in \mathcal{A}_i\}, \ \forall i \in I^+. \end{array}\right.$$

They both are fix-point or saddle-point optimization problems. The MDP problem can be cast as a linear program; see next page.

The Equivalent LP Formulation for MDP

This model can be reformulated as an LP:

$$\text{maximize}_{\mathbf{y}} \quad \boxed{\sum_{i=1}^{m} y_i}$$

$$\text{subject to} \quad \left\{ \begin{array}{rcl} y_1 - \gamma \mathbf{p}_j^T \mathbf{y} & \leq & c_j, \; j \in \mathcal{A}_1 \\ & \vdots & \\ y_i - \gamma \mathbf{p}_j^T \mathbf{y} & \leq & c_j, \; j \in \mathcal{A}_i \\ & \vdots & \\ y_m - \gamma \mathbf{p}_j^T \mathbf{y} & \leq & c_j, \; j \in \mathcal{A}_m. \end{array} \right.$$

Theorem 1 *When $\mathbf{y}$ is maximized, there must be at least one inequality constraint in $\mathcal{A}_i$ that becomes equal for every state i, that is, maximal $\mathbf{y}$ is a fixed point solution.*

The Maze Runner Example

The Fixed-Point formulation:

$$\begin{aligned}
y_0 &= \min\{0+\gamma y_1, 0+\gamma(0.5y_2+0.25y_3+0.125y_4+0.125y_5)\}\\
y_1 &= \min\{0+\gamma y_2, 0+\gamma(0.5y_3+0.25y_4+0.25y_5)\}\\
y_2 &= \min\{0+\gamma y_3, 0+\gamma(0.5y_4+0.5y_5)\}\\
y_3 &= \min\{0+\gamma y_4, 0+\gamma y_5\}\\
y_4 &= 1+\gamma y_5\\
y_5 &= 0 \;(\text{or } y_5 = 0+\gamma y_5)
\end{aligned}$$

The LP formulation:

$$\text{maximize}_{\mathbf{y}} \quad y_0+y_1+y_2+y_3+y_4+y_5$$

subject to change each equality above into inequality

The Interpretations of the LP Formulation

The LP variables $\mathbf{y} \in \mathbf{R}^m$ represent the expected present cost-to-go values of the m states, respectively, for a given policy.

The LP problem entails choosing variables in $\mathbf{y}$, one for each state i, that maximize $\mathbf{e}^T\mathbf{y}$ so that it is the fixed point

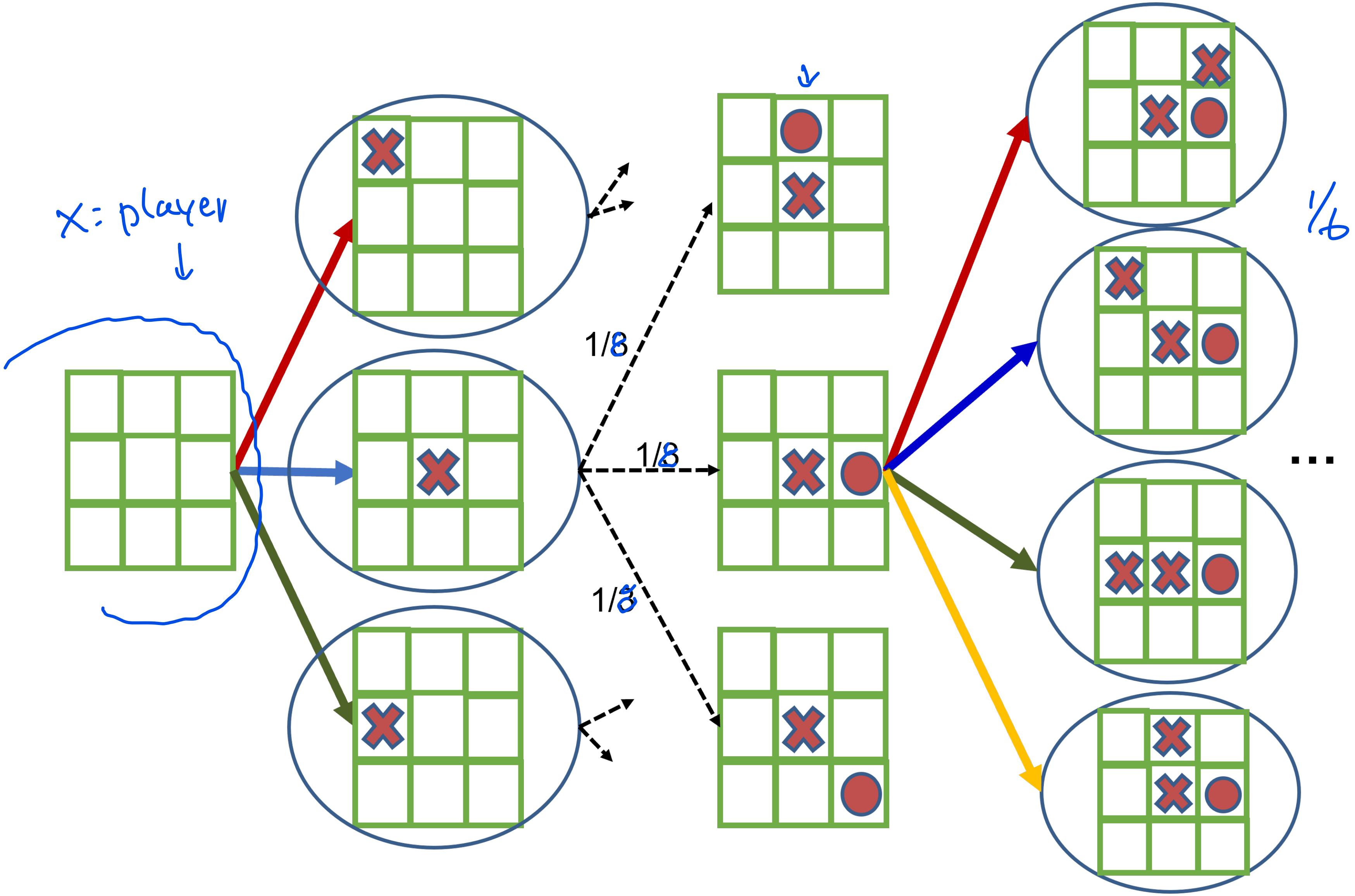
$$y_i^* = \min_{j \in \mathcal{A}_i} \{\mathbf{c}_{j_i} + \gamma \mathbf{p}_{j_i}^T \mathbf{y}\}, \ \forall i,$$

with an optimal policy

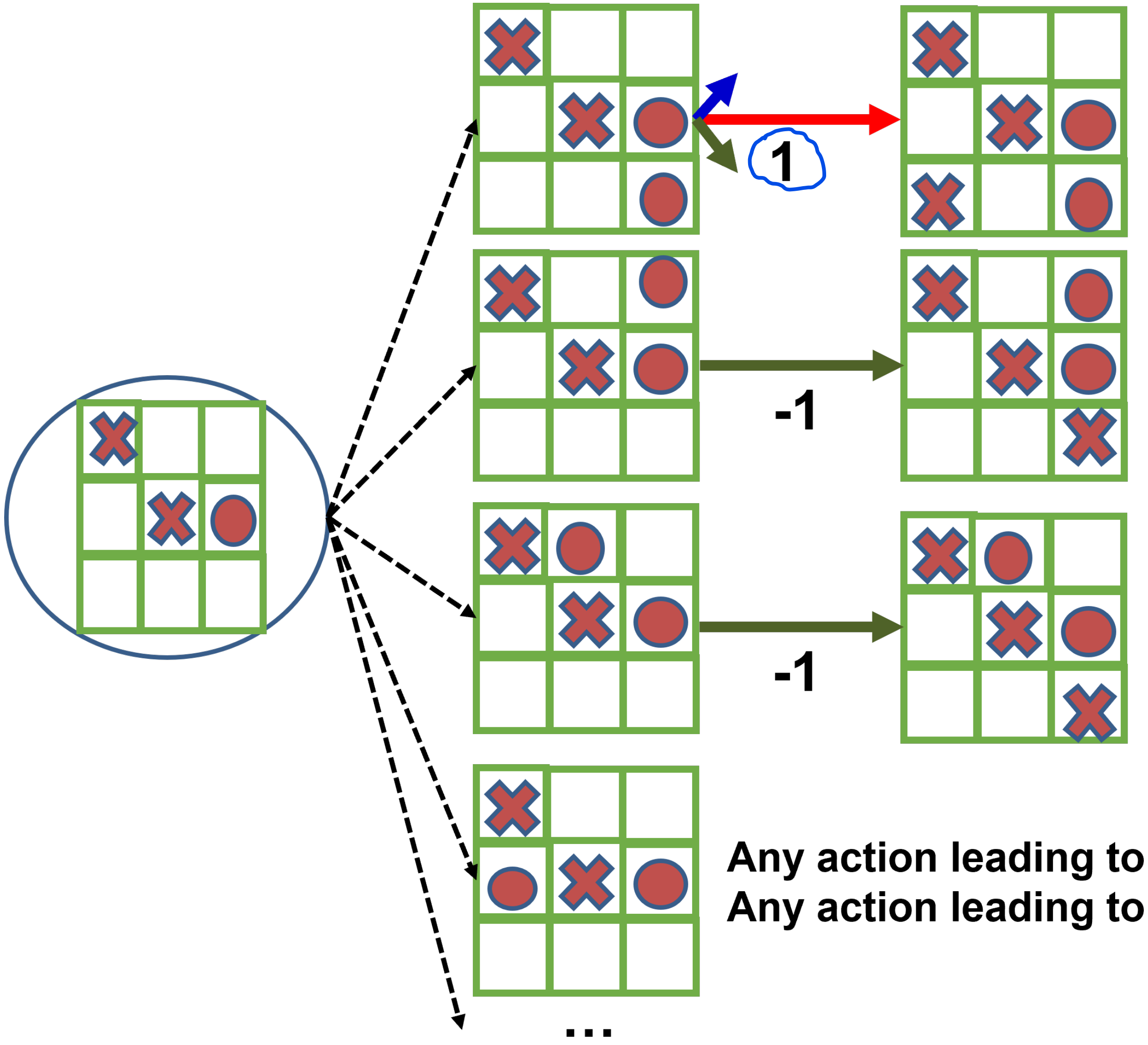
$$\pi_i^* = \arg\min\{\mathbf{c}_j + \gamma \mathbf{p}_j^T \mathbf{y}, \ j \in \mathcal{A}_i\}, \ \forall i.$$

It is well known that there exist a unique optimal stationary policy value vector $\mathbf{y}^*$ where, for each state i, y_i^* is the minimum expected present cost that an individual in state i and its progeny can incur.

States/Actions in the Tic-Tac-Toe Game against a Random Player



Action Costs in the Tic-Tac-Toe Game against a Random Player



Any action leading to win has cost -1
Any action leading to lose has cost 1

States/Actions in the Tic-Tac-Toe Game against an Adversary Player?