第9章 人工智能博弈与安全

南京信息工程大学
计算机学院

应龙

2025年秋季

主要内容

1. Counterfactual Regret Minimization

2. 博弈规则设定*

3. 非完全信息博弈的实际应用*

4. AI and Information Security

Bibliography:
[1] 吴飞 编著, “人工智能导论：模型与算法” (2020), 高等教育出版社. Ch 8, Ch 6.5.4
[2] Stuart J. Russel, Peter Norvig, “Artificial Intelligence: A Modern Approach” (4th Ed. 2020); 中译版 “人工智能 现代方法 (第4版)”, 人民邮电出版社, 2022. Ch 18, Ch 27.3

主要内容

1. Counterfactual Regret Minimization

2. 博弈规则设定*

3. 非完全信息博弈的实际应用*

4. AI and Information Security

Bibliography:
[1] 吴飞 编著, “人工智能导论：模型与算法” (2020), 高等教育出版社. Ch 8
[2] Stuart J. Russel, Peter Norvig, “Artificial Intelligence: A Modern Approach” (4th Ed. 2020); 中译版 “人工智能 现代方法 (第4版)”, 人民邮电出版社, 2022. Ch 18

博弈论与计算机科学

博弈论与计算机科学的交叉领域非常多

- 理论计算机科学：算法博弈论
- 人工智能：多智能体系统、AI游戏参与者、人机交互、机器学习、广告推荐
- 互联网：互联网经济、共享经济
- 分布式系统：区块链

人工智能与博弈论相互结合，形成了两个主要研究方向

- 博弈策略的求解
- 博弈规则的设计

人工智能与博弈论

- 人工智能与博弈论：博弈策略求解
  - 动机
    - 博弈论提供了许多问题的数学模型
    - 纳什存在性定理确定了博弈过程问题存在解
    - 人工智能的方法可用来求解均衡局面或者最优策略
  - 主要问题

    如何高效求解博弈参与者的策略以及博弈的均衡局势？
  - 应用领域
    - 大规模搜索空间的问题求解：围棋
    - 非完美信息博弈问题求解：德州扑克
    - 网络对战游戏智能：Dota、星球大战
    - 动态博弈的均衡解：厂家竞争、信息安全

[1] Matej Moravcik, Martin Schmid, Karel Ha, Milan Hladikand Stephen J. Gaukrodger. "*Refining Subgames in Large Imperfect Information Games*." AAAI Conference on Artificial Intelligence (2016).

[2] Noam Brown, and Tuomas Sandholm. "*Safe and nested subgame solving for imperfect-information games*." Advances in neural information processing systems 30 (2017).

Regret Minimization

➢ 一些定义：

- 对于一个有N个参与者参加的博弈，参与者i在博弈中采取的策略记为σ_i。
- 对于任何序贯决策的博弈对抗，可将博弈过程表示成一棵博弈树。博弈的规则、参与者采取的历史行动、以及博弈当前的状态，组成了信息集 $I_i \in \xi_i$。
- 对于每个信息集 $I_i \in \xi_i$, $\sigma_i(I_i): A(I_i) \to [0,1]$ 是在动作集 $A(I_i)$ 的概率分布函数。参与者 i 的策略空间用 Σ_i 表示。
- 一个策略组包括所有参与者的策略：$\sigma = (\sigma_1, \sigma_2, \dots, \sigma_{|N|})$
- σ_{-i} 表示 σ 中除了 σ_i 之外的所有参与者的策略，即除去参与者 i 所采用的策略。
- 在博弈对决中，不同参与者在不同时刻会采取相应策略以及行动。按策略 σ 对应的行动序列 h 发生的概率表示为 $\pi^{\sigma}(h) = \prod_{i \in N} \pi_i^{\sigma}(h)$, 这里 $\pi_i^{\sigma}(h)$ 表示 参与者 i 使用策略 σ_i 促使行动序列 h 发生的概率。除参与者 i 以外，其他参与者通过各自策略促使行动序列 h 发生的概率可表示为： $\pi_{-i}^{\sigma}(h) = \prod_{j \in N\{i\}} \pi_j^{\sigma}(h)$
- 对于每个参与者 $i \in N$, u_i: $\mathcal{Z} \to R$ 表示参与者 i 的收益函数，即在到达终止状态的序列集合 $\mathcal{Z}$ 中某个序列时，参与者 i 所得到的收益。
- 参与者 i 在给定策略 σ 下所能得到的期望收益为： $u_i(\sigma) = \prod_{h \in \mathcal{Z}} u_i(h) \pi^{\sigma}(h)$

收益函数

Regret Minimization

➢ 最优反应策略

在给定其他参与者的策略组合 σ_{-i} 的情况下，参与者 i 的**最优反应策略** σ_i^* 满足如下条件：

$$u_i(\sigma_i^*, \sigma_{-i}) \geq \max_{\sigma_i' \in \Sigma_i} u_i(\sigma_i', \sigma_{-i})$$

Σ_i 是参与者 i 可以选择的所有策略。当参与者 i 采用最优反应策略时，能够获得最大收益。

在策略组 σ 中，如果每个参与者的策略相对于其他参与者的策略而言都是最佳反应策略，那么策略组σ就是一个纳什均衡 (Nash equilibrium) 策略。在有限对手、有限策略情况下，混合策略的纳什均衡一定存在。

策略组 $\sigma^* = \{\sigma_1^*, \sigma_2^*, \dots, \sigma_N^*\}$ 是纳什均衡当且仅当对每个参与者 $i = 1, \dots, N$, 满足如下条件：

$$u_i(\sigma^*) \geq \max_{\sigma_i' \in \Sigma_i} \mu_i(\sigma_1^*, \sigma_2^*, \dots, \sigma_i', \dots, \sigma_N^*)$$

ε-纳什均衡：

对于给定的正实数ε，策略组 σ 是 ε −纳什均衡当且仅当对于每个参与者 $i \in N$，满足如下条件：

$$u_i(\sigma) + \varepsilon \geq \max_{\sigma_i' \in \Sigma_i} u_i(\sigma_i', \sigma_{-i})$$

纳什均衡松弛解

Regret Minimization

➢ 累加遗憾值

遗憾最小化算法是一种根据以往博弈过程中所得遗憾程度来选择未来行为的方法。

参与者 i 在过去 T 轮中采取策略 σ_i 的累加遗憾值定义如下：

$$Regret_i^T(\sigma_i) = \sum_{t=1}^{T}(u_i(\sigma_i, \sigma_{-i}^t) - u_i(\sigma^t))$$

除参与者 i 以外，其他参与者的策略保持不变

其中 σ^t 和 σ_{-i}^t 分别表示第 t 轮中所有参与者的策略组合和除了参与者 i 以外的策略组合。简单地说，累加遗憾值代表着在过去 T 轮中，参与者 i 在每一轮中选择策略 σ_i 所得收益与采取其他策略所得收益之差的累加。

在得到参与者 i 的所有可选策略的遗憾值后，可以根据遗憾值的大小来选择后续第 $T+1$ 轮博弈的策略，这种选择方式被称为遗憾匹配 [Greenwald 2006]。

Regret Minimization

➢ 有效遗憾值

通常遗憾值为负数的策略被认为不能提升下一时刻收益，如下定义有效遗憾值：

$$Regret_i^{T,+}(\sigma_i) = \max(Regret_i^T(\sigma_i), 0)$$

利用有效遗憾值的遗憾匹配可得到参与者 i 在 T 轮后第 $T+1$ 轮选择策略 σ_i 的概率$P(\sigma_i^{T+1})$为：

$$P(\sigma_i^{T+1}) = \begin{cases} \dfrac{Regret_i^{T,+}(\sigma_i)}{\sum_{\sigma_i' \in \Sigma_i} Regret_i^{T,+}(\sigma_i')} & if \sum_{\sigma_i' \in \Sigma_i} Regret_i^{T,+}(\sigma_i') > 0 \\ \dfrac{1}{|\Sigma_i|} & \text{otherwise} \end{cases}$$

$|\Sigma_i|$表示参与者i所有策略的总数。

如果在过往T轮中策略σ_i所带来的遗憾值大、其他策略σ_i'所带来的遗憾值小，则在第$T+1$轮选择策略σ_i的概率值$P(\sigma_i^{T+1})$就大。也就是说，带来越大遗憾值的策略具有更高的价值，因此其在后续被选择的概率就应该越大。如果没有一个能够提升前T轮收益的策略，则在后续轮次中随机选择一种策略。

依照一定的概率选择行动是为了防止对手发现自己所采取的策略（如采取遗憾值最大的策略）。

Regret Minimization

➢ 石头-剪刀-布 (Rock-Paper-Scissors)

假设两个参与者A和B进行石头-剪刀-布（Rock-Paper-Scissors，RPS）的游戏，获胜参与者收益为1分，失败参与者收益为-1分，平局则两个参与者收益均为0分

	石头(R)	剪刀(S)	布(P)
石头(R)	$(0,0)$	$(1,-1)$	$(-1,1)$
剪刀(S)	$(-1,1)$	$(0,0)$	$(1,-1)$
布(P)	$(1,-1)$	$(-1,1)$	$(0,0)$

第1局，若参与者A出石头（R），参与者B出布（P），则此时参与者A的收益 $\mu_A(R,P)=-1$，参与者B的收益为 $\mu_B(P,R)=1$

对于参与者A来说，在参与者B出布（P）这个策略情况下，如果参与者A选择出布（P）或者剪刀（S），则参与者A对应的收益值 $\mu_A(P,P)=0$或者 $\mu_A(S,P)=1$

所以第1局之后，参与者A没有出布的遗憾值为 $\mu_A(P,P)-\mu_A(R,P)=0-(-1)=1$，没有出剪刀的遗憾值为 $\mu_A(S,P)-\mu_A(R,P)=1-(-1)=2$

所以在第2局中，参与者A选择石头、剪刀和布这三个策略的概率分别为 0，$^2/_3$，$^1/_3$。因此，参与者A趋向于在第二局中选择出剪刀这个策略

Regret Minimization

➢ 石头-剪刀-布的例子

参与者i每一轮悔值计算公式：$\mu_i(\sigma_i, \sigma_{-i}^t) - \mu_i(\sigma^t)$

在第一轮中参与者A选择石头和参与者B选择布、在第二局中参与者A选择剪刀和参与者B选择石头情况下，则参与者A每一轮遗憾值及第二轮后的累加遗憾取值如下：

每轮悔值\策略	石头	剪刀	布
第一轮悔值	0	2	1
第二轮悔值	1	0	2
$Regret_A^2$	1	2	3

- 从上表可知，在第三局时，参与者A选择石头、剪刀和布的概率分别为$1/6$、$2/6$、$3/6$
- 在实际使用中，可以通过多次模拟迭代累加遗憾值找到每个参与者在每一轮次的最优策略
- 但是当博弈状态空间呈指数增长时，对一个规模巨大的博弈树无法采用最小遗憾算法，因此需要采取 反事实遗憾最小化算法 (counterfactual regret minimization)

Counterfactual Regret Minimization

- Counterfactual Regret Minimization (CFR)

如果不能遍历计算所有节点的遗憾值，那么可以采用反事实遗憾最小化算法(Counterfactual Regret Minimization, CFR) 来进行模拟计算。

参与者 i 每一轮悔值计算公式：$\mu_i(\sigma_i, \sigma_{-i}^t) - \mu_i(\sigma^t)$

假设：

- 集合A是博弈中所有参与者所能采用的行为集（如在石头-剪刀-布游戏中出石头、出剪刀或出布三种行为）
- I 为信息集，博弈树的中间结点，包含了该中间结点的状态以及到达该结点的所有的历史行动，在信息集 I 下所能采取的行为集合记为 $A(I)$

参与者 i 在第 t 轮次采取的行动 $a_i \in A(I_i)$ 反映了在该轮次所采取的策略 σ_i^t。包含参与者 i 在内的所有参与者在第 t 轮次采取的行动 $a \in A(I)$ 构成了一组策略组合 σ^t。

在信息集 I 下采取行动 a 所反映的策略记为 $\sigma_{I \to a}$。

Counterfactual Regret Minimization

- 所有参与者交替采取的行动序列记为h（从根节点到当前节点的路径），对于所有参与者的策略组合σ，行动序列h出现的概率记为$\pi^{\sigma}(h)$。
- 不同的行动序列可以从根节点到达当前节点的信息集I。在策略组合σ下，所有能够到达该信息集的行动序列的概率累加就是该信息集的出现概率，即$\pi^{\sigma}(I) = \sum_{h \in I} \pi^{\sigma}(h)$。
- 博弈的终结局势集合也就是博弈树中叶子节点的集合，记为Z。对于任意一个终结局势 $z \in Z$，参与者i在此终点局势下的收益记作 $u_i(z)$。给定行动序列h，依照策略组合σ最终到达终结局势z的概率记作$\pi^{\sigma}(h, z)$。

Counterfactual Regret Minimization

在策略组合σ下，对参与者i而言，如下计算从根节点到当前节点的行动序列路径 h 的虚拟价值（注：行动序列 h 未能使博弈进入终结局势）：

$$v_i(\sigma, h) = \sum_{z \in Z} \underbrace{\pi_{-i}^{\sigma}(h)}_{\text{不考虑参与者 } i \text{ 的策略到达当前节点概率}} \times \underbrace{\pi^{\sigma}(h, z)}_{\text{从当前结点到叶子结点的概率}} \times \underbrace{u_i(z)}_{\text{叶子结点 } z \text{ 的收益}}$$

$\pi_{-i}^{\sigma}(h)$: 不考虑参与者 i 的策略，仅考虑其他参与者策略而经过路径 h 到达当前节点的概率。即使参与者 i 有其他策略，总是要求参与者 i 在每次选择时都选择路径 h 中对应的动作，以保证从根节点出发能够到达当前节点。

参与者 i 基于路径 h 到达当前节点并采取行动 a 的 Counterfactual Regret：

$$r_i(h, a) = v_i(\sigma_{I \to a}, h) - v_i(\sigma, h)$$

信息集 I 的遗憾值： $r_i(I, a) = \sum_{h \in I} r_i(h, a)$

反事实遗憾最小化的遗憾值是T轮重复博弈后的累加值：

$$Regret_i^T(I, a) = \sum_{t=1}^{T} r_i^t(I, a)$$

参与者 i 在第 t 轮中于当前节点选择行动 a 的遗憾值。

Counterfactual Regret Minimization

➢ 有效反事实遗憾值

进一步可以定义有效反事实遗憾值：

$$Regret_i^{T,+}(I,a) = \max\left(R_i^T(I,a), 0\right)$$

根据有效反事实遗憾值进行遗憾匹配以计算经过T轮博弈后，参与者i在信息集I情况下于后续$T+1$轮选择行动a的概率：

$$\sigma_i^{T+1}(I,a) = \begin{cases} \dfrac{Regret_i^{T,+}(I,a)}{\sum_{a\in A(I)} Regret_i^{T,+}(I,a)} & \text{if } \sum_{a\in A(I)} Regret_i^{T,+}(I,a) > 0 \\ \dfrac{1}{|A(I)|} & \text{otherwise} \end{cases}$$

Counterfactual Regret Minimization

在反事实遗憾最小化算法的求解过程中，同样需要反复模拟多轮博弈来拟合最佳反应策略，算法步骤如下：

1) 初始化遗憾值和累加策略表；
2) 采用随机选择的方法来决定策略；
3) 利用当前策略与对手进行博弈；
4) 计算每个参与者采取每次行为后的反事实遗憾值；
5) 根据博弈结果计算每个行动的累加遗憾值大小来更新策略；
6) 重复3)到5)步若干次，不断的优化策略；
7) 根据重复博弈最终的策略，完成最终的动作选择。

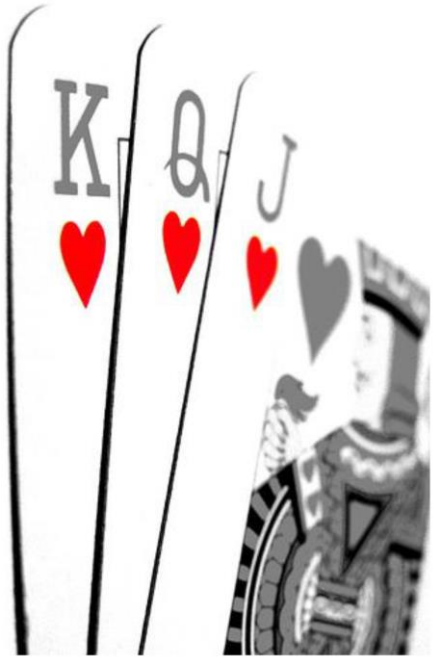
Counterfactual Regret Minimization

➢ 例题 Kuhn's Pocker

库恩扑克[Kuhn 1950]是一种简单的有限注扑克游戏，由两名参与者进行博弈。

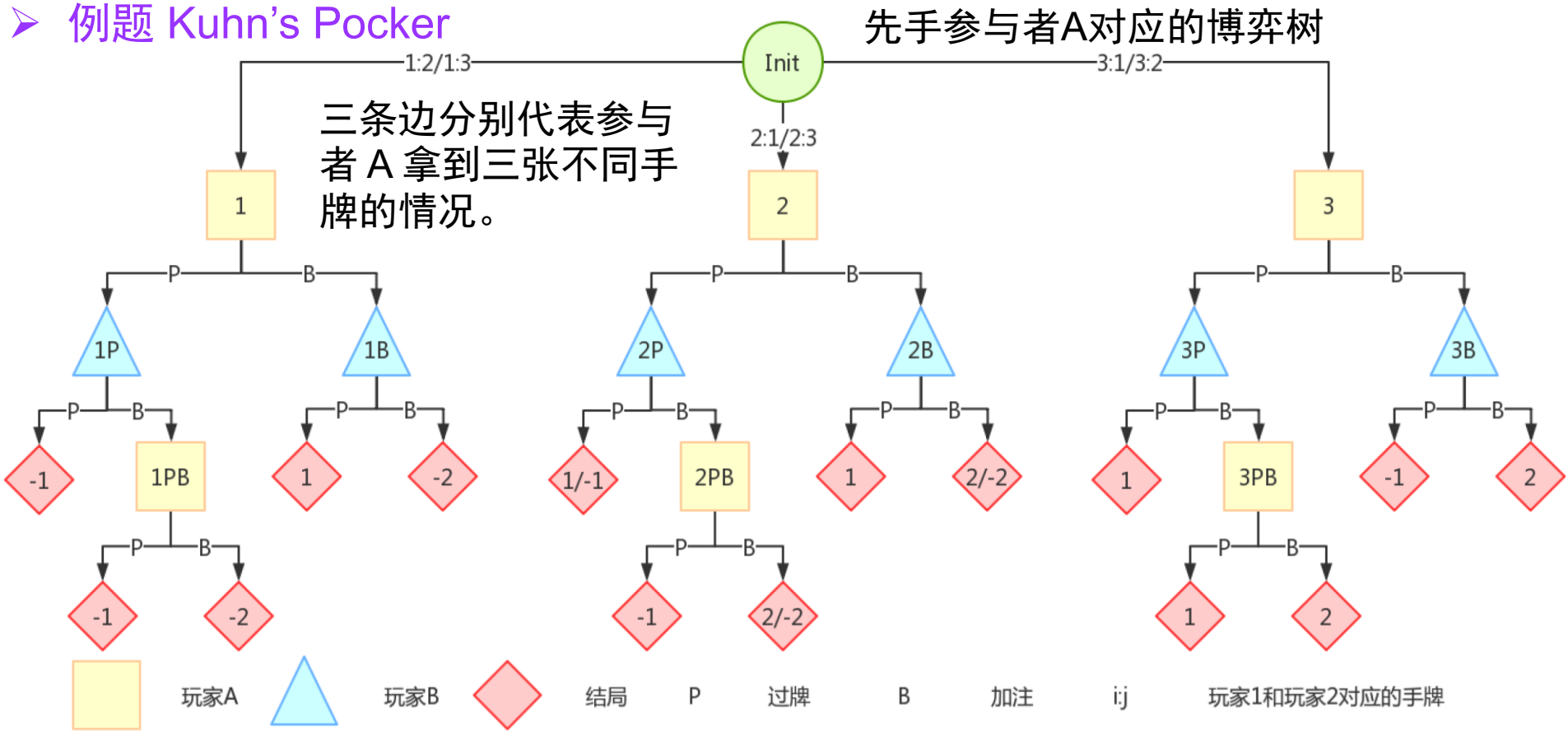
- 游戏中仅提供牌值为1、2和3的三张纸牌。
- 每轮中每位参与者各持一张纸牌，每位参与者根据各自判断来决定是否追加定额赌注。
- 摊牌阶段时来比较未弃牌参与者的底牌大小，底牌值最大的参与者即为胜者。

玩家A	玩家B	玩家A	结果
过牌	过牌	\	牌值大的玩家 +1
加注	加注	\	牌值大的玩家 +2
过牌	加注	过牌	玩家B +1
过牌	加注	加注	牌值大的玩家 +2
加注	过牌	\	玩家A +1



Counterfactual Regret Minimization

➢ 例题 Kuhn's Pocker



库恩扑克的信息集（即博弈树的中间节点）有12个:
$\{1,1P,1B,1BP,2,2P,2B,2PB,3,3P,3B,3PB\}$。在信息集中， P表示过牌、B表示下注。

要注意的是从根节点到达每个信息集的路径只有一条，该路径所对应的行动序列也只有一种，例如信息集 $1PB$（博弈树中的一个中间节点）可通过如下路径到达：

1(参与者A拿到大小为1纸牌) $\xrightarrow{P}$ $1P$(参与者A选择过牌) $\xrightarrow{B}$ $1PB$(参与者B选择加注)

Counterfactual Regret Minimization

➢ 例题 Kuhn's Pocker

计算参与者A通过路径$1 \xrightarrow{P} 1P \xrightarrow{B} 1PB$到达当前节点$\{1PB\}$后选择"**过牌**"行动的遗憾值。

- 在当前策略下，行动序列路径$h = \{P \to B\}$产生的概率：

$$\pi^{\sigma}_{-A}(h) = \pi^{\sigma}_{B}(h) = 1 \times 0.5 = 0.5$$

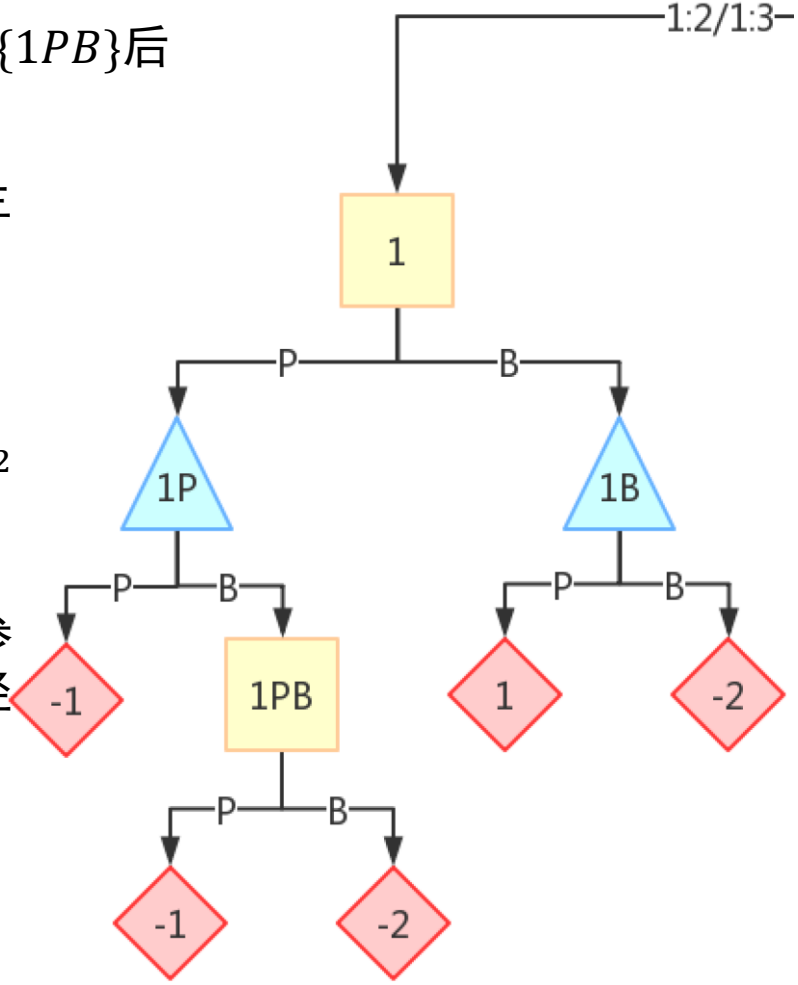
- 在$\{1PB\}$节点选择加注和过牌的概率均为50%，所以当前策略下从当前节点到达终结状态z_1和z_2的概率分别为：

$$\pi^{\sigma}(h, z_1) = 0.5, \pi^{\sigma}(h, z_2) = 0.5$$

- 由于已知 $u_A(z_1) = -1$和 $u_A(z_2) = -2$，可知参与者A在策略σ作用下从根节点出发、经过路径$h = \{P \to B\}$到达当前节点后的虚拟价值：

$$v_A(\sigma, h) = \sum_{z \in Z} \pi^{\sigma}_{-A}(h) \times \pi^{\sigma}(h, z) \times u_A(z)$$

$$= 0.5 \times 0.5 \times (-1) + 0.5 \times 0.5 \times (-2) = -0.75$$

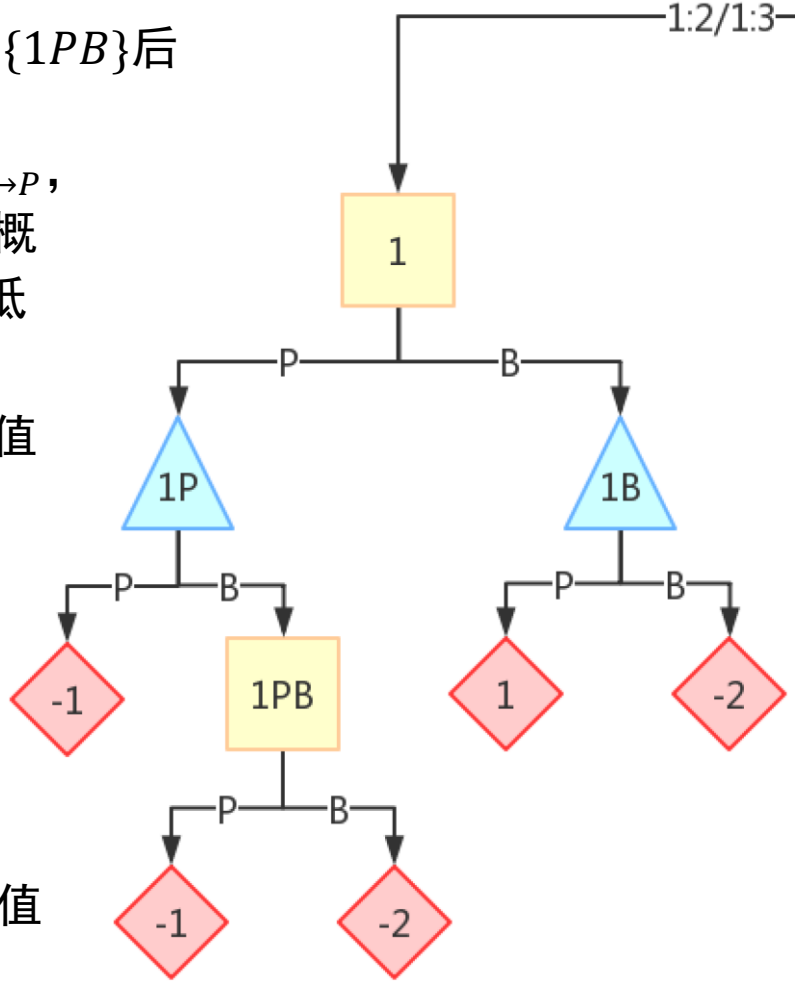


Counterfactual Regret Minimization

➢ 例题 Kuhn's Pocker

计算参与者A通过路径$1 \xrightarrow{P} 1P \xrightarrow{B} 1PB$到达当前节点$\{1PB\}$后选择"**过牌**"行动的遗憾值。

- 若参与者A在 $\{1PB\}$ 节点选择"过牌"行动 $\sigma_{\{1PB\}\to P}$，此时参与者B促使行动序列 $h = \{P \to B\}$发生的概率仍然为 $\pi_B^{\sigma_{\{1PB\}\to P}}(h) = 0.5$。从当前节点出发抵达的终结状态只有$z_1$，所以$\pi^{\sigma_{\{1PB\}\to P}}(h, z_1) = 1$
- 参与者A在当前节点选择"过牌"行动后的虚拟价值为：$v_A(\sigma_{\{1PB\}\to P}, h) =$
$$\pi_{-A}^{\sigma_{\{1PB\}\to P}}(h) \times \pi^{\sigma_{\{1PB\}\to P}}(h, z_1) \times u_A(z_1)$$
$$= 0.5 \times 1 \times (-1) = -0.5$$
- 在信息集$\{1PB\}$上采取"**过牌**"的虚拟遗憾值：
$$r_A(\{1PB\}, P) = r_A(h, P)$$
$$= v_A(\sigma_{\{1PB\}\to P}, h) - v_A(\sigma, h) = 0.25$$
- 在第一轮博弈中，累加虚拟遗憾值与行为遗憾值相同: $Regret_A^1(\{1PB\}, P) = \sum_{t=1}^{1} r_A^t(\{1PB\}, P)$
- 可以计算信息集$\{1PB\}$上采取"**加注**"策略的遗憾值：$Regret_A^1(\{1PB\}, B) = r_A(\{1PB\}, B) = r_A(h, B) = -0.25$



Counterfactual Regret Minimization

➢ 例题 Kuhn's Pocker

库恩扑克的博弈共有12个信息集: $\{1,1P,1B,1BP,2,2P,2B,2PB,3,3P,3B,3PB\}$
通过反复迭代计算，可以得到到达各个信息集应采取行动的概率：

	1	1B	1P	1PB	2	2B	2P	2PB	3	3P	3B	3PB
B	0.13	0	0.33	0	0	0.33	0	0.46	0.41	1	1	1
P	0.87	1	0.67	1	1	0.67	1	0.54	0.59	0	0	0

对参与者A，库恩扑克的混合策略纳什均衡的理论解如下：$\alpha \in [0, 1/3]$

	1	1B	1P	1PB	2	2B	2P	2PB	3	3P	3B	3PB
B	α	\	\	0	0	\	\	$1/3+\alpha$	3α	\	\	1
P	$1-\alpha$	\	\	1	1	\	\	$2/3-\alpha$	$1-3\alpha$	\	\	0

可见，算法得到的解与理论得到的解之间较为接近，验证了算法的有效性。

Counterfactual Regret Minimization

➢ Monte Carlo CFR

MCCFR improves CFR by using **sampling techniques** to reduce the need to traverse the entire game tree, which is computationally prohibitive for large games.

1.Sampling: Instead of computing regrets over the entire game tree, MCCFR computes them on sampled trajectories.

2.Estimators: Regret and utility values are estimated based on the sampled paths.

3.Convergence: By sampling over multiple iterations, MCCFR approximates the results of full CFR while significantly reducing computation.

Sampling Methods in MCCFR

$$v_i(\sigma, h) = \sum_{z \in Z} \underbrace{\pi^{\sigma}_{-i}(h)} \times \underbrace{\pi^{\sigma}(h, z)} \times \underbrace{u_i(z)}$$

1.External Sampling:

- Samples actions for all players except the current one.
- Estimates the regret and utility for the sampled path.
- Reduces complexity but may introduce high variance in large trees.

2.Outcome Sampling:

- Samples a single terminal node (outcome) per iteration.
- Weighs updates by the probability of the sampled path.
- Balances variance and computational cost, commonly used in practice.

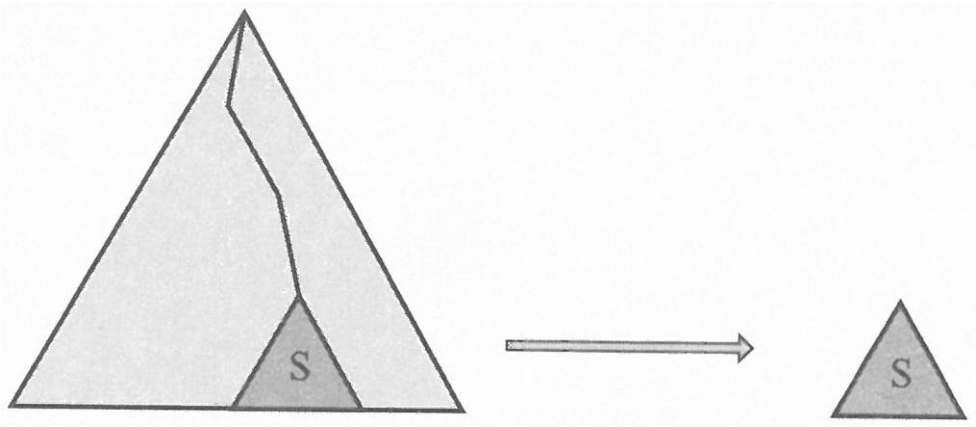
3.Internal Sampling:

- Samples trajectories for the current player's decisions.
- Focuses regret updates on the player's choices.

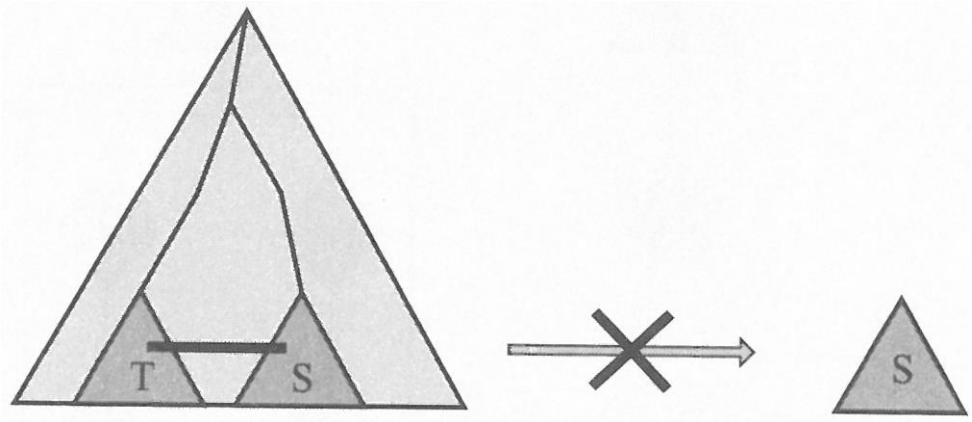
安全子博弈

➢ 子博弈

从当前已经完成的部分博弈出发，将接下来博弈过程视为是一个单独子博弈，然后找到子博弈的最优反应策略，这样可以减小计算量，以便在接近叶节点的情况下得到更加精确的结果。



完全信息下的子博弈与博弈其他部分无关，可以单独考虑



非完全信息下的子博弈与博弈其他部分相关，不能单独考虑

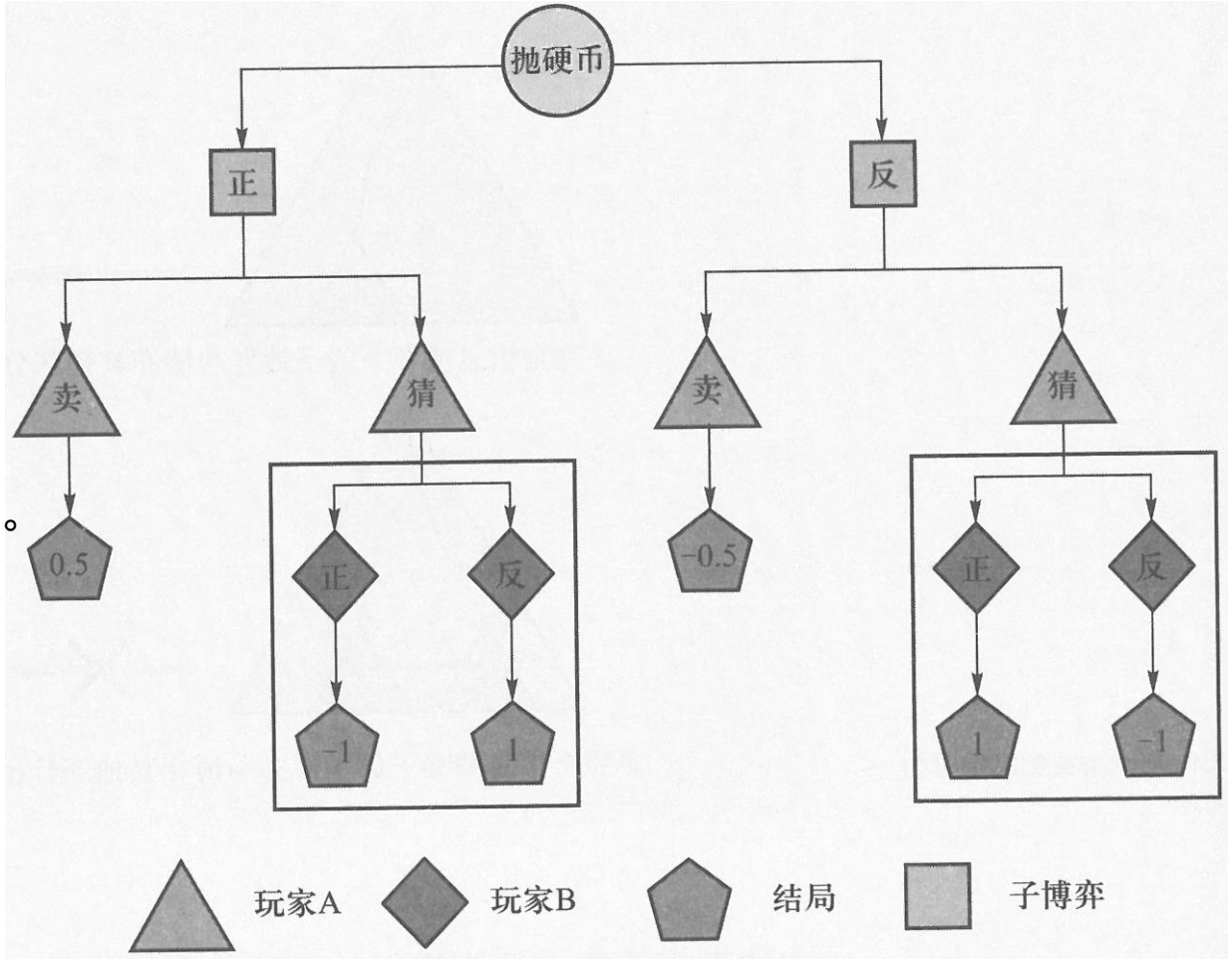
安全子博弈

➢ 抛硬币游戏

玩家A随机抛一个硬币，然后根据硬币的正反面可以决定将硬币卖掉或者让玩家B来猜测硬币的正反面。

如果硬币抛掷结果是正面，玩家A卖掉这一硬币可以获得0.5元收益；如果硬币抛掷结果是反面，则玩家A卖掉这一硬币会亏损0.5元（-0.5元）。

如果玩家A不采取售卖硬币的行动，而是让玩家B猜硬币的正反，则玩家B猜错正反前提下玩家A可以获得1元或者玩家B猜对正反前提下玩家A会亏损1元（-1元）。



对玩家B这是一个不完全信息博弈

安全子博弈

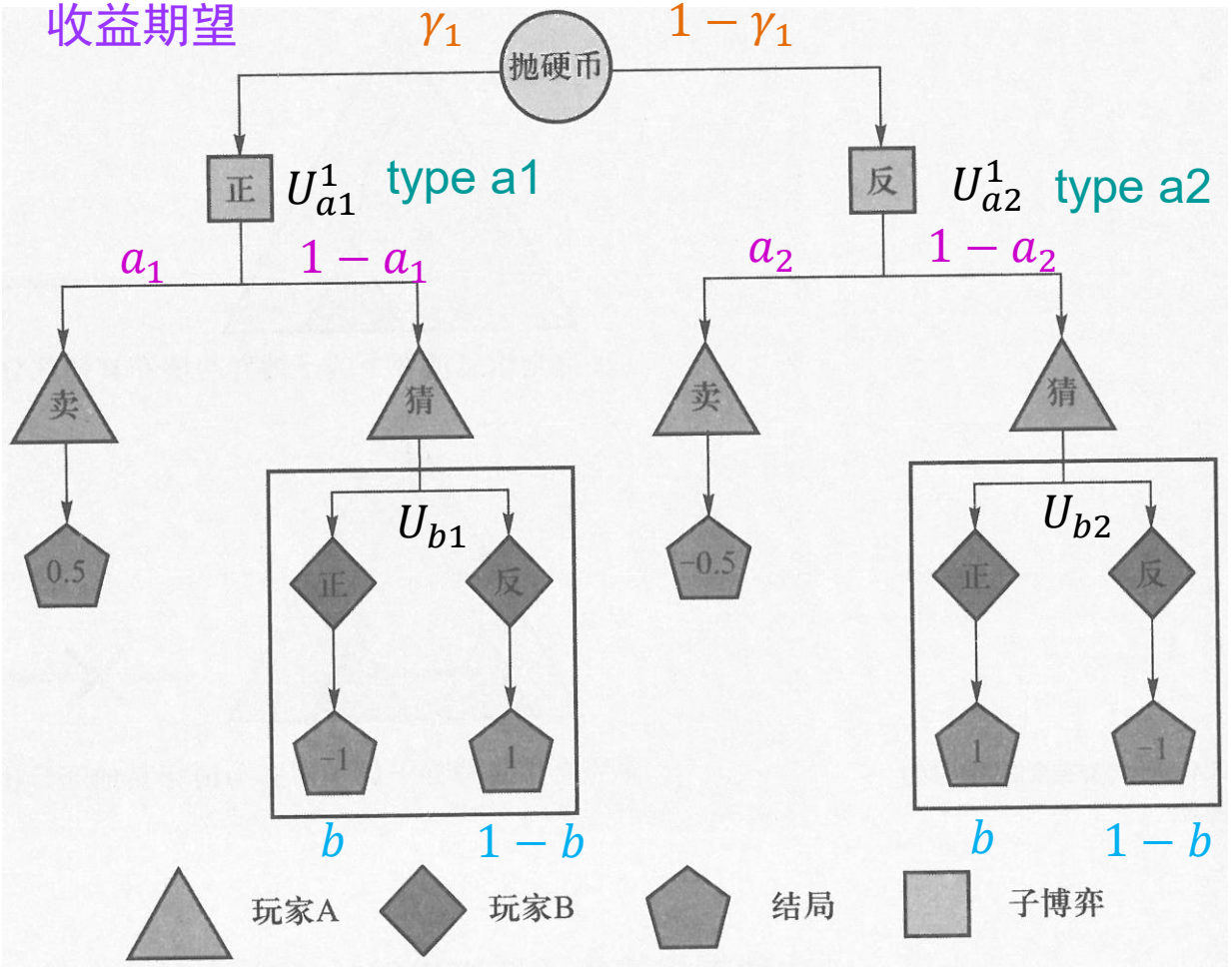
➢ 抛硬币游戏

$$U_{b1}^2 = -b + (1-b) = 1 - 2b$$
$$U_{b2}^2 = b - (1-b) = 2b - 1$$
$$U_{a1}^1 = 0.5a_1 + U_{b1}^2(1-a_1)$$
$$U_{a2}^1 = -0.5a_2 + U_{b2}^2(1-a_2)$$
$$U_a^1 = U_{a1}^1\gamma_1 + U_{a2}^1(1-\gamma_1)$$

⇨ 线性规划问题

$$\nabla_{a_1} U_a^1 = 0.5 - U_{b1}^2 = 0$$
$$\nabla_{a_2} U_a^1 = -0.5 - U_{b2}^2 = 0$$
$$b = 0.25$$

在玩家B知道玩家A卖掉硬币所获得收益的情况下，对玩家B来说，他的纳什均衡策略是不论玩家A选择卖掉硬币还是让其进行猜测，玩家A的收益都一样。



纳什均衡的上界

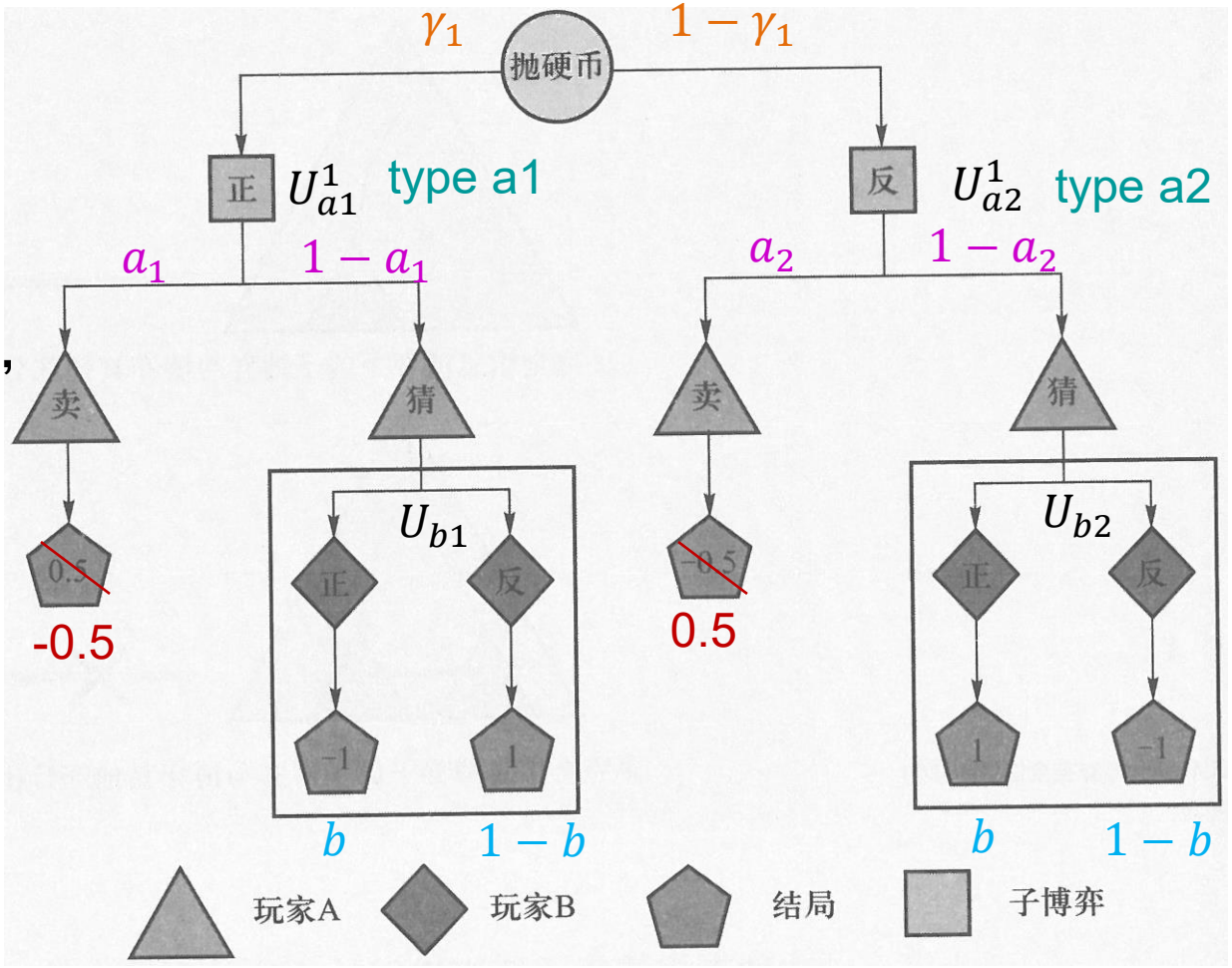
在这场博弈中，即使玩家A选择了让玩家B来猜测硬币投掷结果的正反，玩家B也必须根据玩家A选择卖掉硬币的期望收益来计算自己纳什均衡的策略。

安全子博弈

➢ 抛硬币游戏

假设玩家A卖掉硬币的收益发生变化如图所示，可以求出玩家B的纳什均衡反应策略应该是以0.75概率来猜测硬币投掷结果是正面。

如果单独考虑猜硬币的情况，而不考虑卖硬币的情况，在正面和反面出现概率相同的情况下，玩家B应该以0.5概率来猜测硬币投掷结果是正面。



安全子博弈

一个不完全信息博弈被形式化为一个 贝叶斯博弈：

$$G = \left(N,\ \{A_i\}_{i\in N},\ \{T_i\}_{i\in N},\ p(t),\ \{u_i(\cdot,t)\}_{i\in N}\right)$$

1 参与者 (Players)

$N = (1, \ldots, n)$

2 类型（Types）

- 每个玩家 i 有一个类型 $t_i \in T_i$
- 类型包含该玩家的**私人信息**
  - 偏好参数
  - 成本函数
  - 信号 / 状态信息

3 先验分布(Common Prior)

联合分布：$p(t_1, \ldots, t_n)$

共同知识：不同玩家的“主观信念”是条件分布：

$p(t_{-i}|t_i)$

4 策略(Strategies)

在不完全信息下，策略是：

从类型到行动的映射

$s_i: T_i \to A_i$

而不是单一动作

5 期望效用

玩家 i 的目标是最大化 **条件期望效用**：

$$\mathbb{E}_{t_{-i}|t_i}\left[u_i(s_i(t_i), s_{-i}(t_{-i}), t_i, t_{-i})\right]$$

标准解概念：贝叶斯纳什均衡（BNE）

一组策略$\{s_i^*\}$ 构成贝叶斯纳什均衡，如果对所有参与者i、所有类型t_i

$$s_i^*(t_i) \in \arg\max_{a_i \in A_i} \mathbb{E}_{t_{-i}|t_i}\left[u_i(a_i, s_{-i}^*(t_{-i}), t)\right]$$

安全子博弈

对于一个非完全信息的博弈，通常可以找到一些从全局出发的近似解法，如使用反事实遗憾值最小化 (CFR) 算法。**安全子博弈**是指在子博弈的求解过程中，得到的结果一定不差于全局的近似解法。

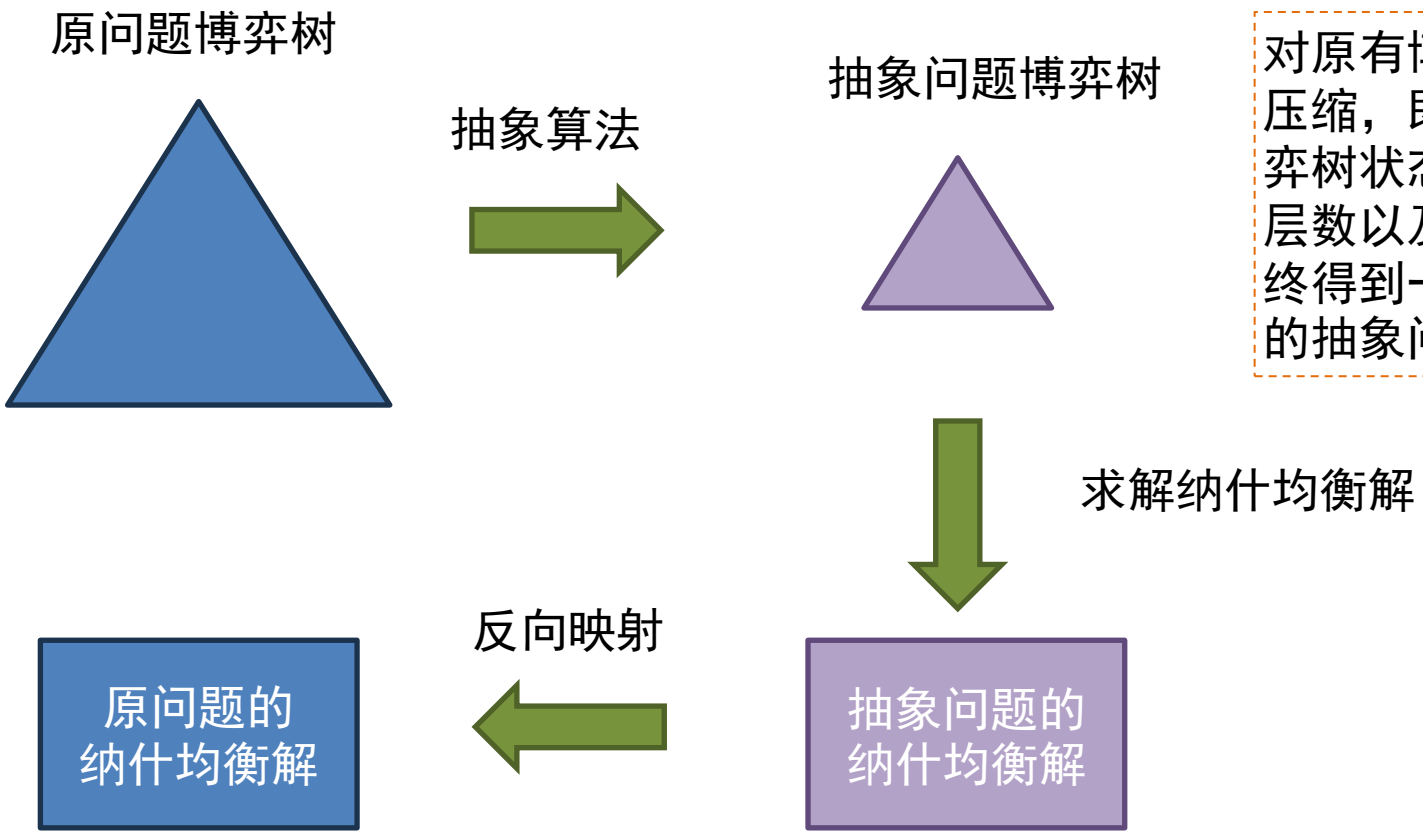
- 在猜硬币的过程中，如果参与者B的行动是通过预测参与者A卖掉硬币的收益来决定，猜硬币这个子博弈就是安全的。反之如果参与者B的行动仅考虑猜硬币的过程（本例中为不考虑A买硬币的收益），则猜硬币这个子博弈就是不安全的。

可以通过虚拟价值的估计，来找到任意一个博弈的安全子博弈。

- 给定双人博弈树中的一个信息集 I_1，假设参与者1在信息集 I_1' 采取行动 a' 可到达信息集 I_1。如果在信息集 I_1' 采取行动 a^* 进入另外一个信息集I_{1c}可得到更大的价值，则意味着参与者1所采取的行动只获得了局部最优解(在信息集 I_1)。
- 可将采取行动 a^* 和行动 a' 所分别获得价值之间的差距作为奖励，增加到信息集 I_1 的价值评估中。
- 已经证明，只要信息集的奖励足够小，那么信息集 I_1 所对应的子博弈就是安全的[Brown et al., 2018]。可以通过信息集 I_1 和信息集 I_1' 虚拟价值的下限评估行动 a^* 和行动 a' 价值的差距。

非完全信息博弈的实际应用

- 求解非完全信息博弈纳什均衡的一般方法



此外，深度强化学习越来越多地被应用于博弈的策略求解。强化学习的主要思想是通过试错和搜索来优化自身的行动[13]，这一过程可以直接被应用于博弈的决策中。

主要内容

1. Counterfactual Regret Minimization

2. 博弈规则设定*

3. 非完全信息博弈的实际应用*

4. AI and Information Security

Bibliography:

- ✓ 吴飞 编著, “人工智能导论：模型与算法” (2020), 高等教育出版社. Ch 8
- ✓ Stuart J. Russel, Peter Norvig, “Artificial Intelligence: A Modern Approach” (4th Ed. 2020); 中译版 “人工智能 现代方法” (4rd Ed., 2022), 清华大学出版社. Ch 18

博弈论机制设计

◆ 人工智能与博弈论：博弈机制设计

- 动机：个体理性可能导致群体非理性。假设博弈的参与者足够理性，应该如何设计博弈规则或者使得博弈的最终局势能尽可能达到整体利益的最大化，或者保证博弈的公正性？

  例如，许多城市为了避免机动车过快增长造成城市拥堵，因此限制了车牌的发放量。这样，车牌的发放方式就是需要决策者经过周密的考虑后决定的。通常而言，车牌的发放既要能满足有紧迫需求的人，又要满足普通家庭的日常需求，所以很多城市都采取了车牌竞价和随机摇号两种方式发放车牌。

- 挑战：规则复杂，计算量大
- 利用人工智能的算法和思想可以为许多博弈规则设计问题提供帮助：
  - 拍卖竞价：互联网广告投放、车牌竞价
  - 供需匹配：污染权、学校录取
  - 公正选举：选举制度、表决制度、议席分配

博弈规则设计：双边匹配算法

- 双边匹配问题：在生活中，人们常常会碰到与资源匹配相关的决策问题(如求职就业、报考录取等)，这些需要双向选择的情况被称为是双边匹配问题。在双边匹配问题中，需要双方互相满足对方的需求才会达成匹配。

➢ 稳定婚姻问题 (stable marriage problem)

给定成员偏好顺序的情况下，为两组成员寻找稳定的匹配。

假设有n个单身男性构成的集合$M = \{m_1, m_2, \dots, m_n\}$，以及$n$个单身女性构成的集合$F = \{f_1, f_2, \dots, f_n\}$:

- 对于任意一名单身男性 m_i，都有自己爱慕的单身女性的顺序 $s_{m_i} \coloneqq f_{m_{i,1}} \succ f_{m_{i,2}} \succ \cdots \succ f_{m_{i,n}}$, $f_{m_{i,j}}$ 表示第i名男性所喜欢单身女性中排在第j位的单身女性；
- 同理对于任意一名单身女性 f_i，也有其爱慕的单身男性顺序 $s_{f_i} \coloneqq m_{f_{i,1}} \succ m_{f_{i,2}} \succ \cdots \succ m_{f_{i,n}}$, $m_{f_{i,j}}$ 表示第i名女性所喜欢单身男性中排在第j位的单身男性。
- 算法的最终目标是为这$2n$个男士和女士匹配得到n对伴侣，每一对伴侣可以表示为(m_i, f_j)。

双向奔赴

稳定解：不存在未达成匹配的两个人都更倾向于选择对方而不是自己当前的匹配对象。

不稳定婚姻匹配：存在不是伴侣的一男和一女，彼此可接受对方，而且要么单身，要么对自己目前伴侣的好感度没有对方高。

匹配的稳定是指没有任何人能从偏离稳定状态中获益。如果将匹配问题看做是一种合作博弈的话，稳定状态解就是纳什均衡解。

博弈规则设计：双边匹配算法

- 尝试修补策略

假设有4名单身男性{1,2,3,4}和4名单身女性$\{A,B,C,D\}$，他（她）们的爱慕序列如表所示

男性	偏好	女性	偏好
1	$A \succ D \succ C \succ B$	A	$3 \succ 4 \succ 2 \succ 1$
2	$A \succ B \succ C \succ D$	B	$3 \succ 2 \succ 4 \succ 1$
3	$A \succ C \succ D \succ B$	C	$1 \succ 3 \succ 4 \succ 2$
4	$B \succ A \succ D \succ C$	D	$2 \succ 4 \succ 3 \succ 1$

先对所有人按照一男一女随机匹配，如果出现不稳定的匹配，则马上重新匹配，没有匹配在一起的两人重新组成伴侣，最后直到不存在不稳定的匹配。表面上看，这种“修补”策略一定能得到稳定的匹配结果，但是在“修补”的过程中很容易陷入死循环。

匹配1	修补	匹配2	修补	匹配3	修补	匹配4	修补	匹配5
$(1,A)$	$(2,A)$	$(1,B)$	$(4,A)$	$(1,B)$	$(4,B)$	$(1,A)$	$(2,B)$	$(1,A)$
$(2,B)$		$(2,A)$		$(2,D)$		$(2,D)$		$(2,B)$
$(3,C)$		$(3,C)$		$(3,C)$		$(3,C)$		$(3,C)$
$(4,D)$		$(4,D)$		$(4,D)$		$(4,B)$		$(4,D)$

博弈规则设计：双边匹配算法

- Gale-Shapely 算法 (G-S 算法)

美国数学家大卫·盖尔和博弈论学家沙普利提出了针对双边稳定匹配问题的解决算法（也被称为Gale- Shapely算法或G-S算法），并将其应用于稳定婚姻问题的求解[Gale 1962]，算法过程如下：

- **单身男性向最喜欢的女性表白**
- **所有收到表白的女性从向其表白男性中选择最喜欢的男性，暂时匹配**
- **未匹配的男性继续向没有拒绝过他的女性表白。收到表白的女性如果没有完成匹配，则从这一批表白者中选择最喜欢男性。即使收到表白的女性已经完成匹配，但是如果她认为有她更喜欢的男性，则可以拒绝之前的匹配者，重新匹配**
- **如此循环迭代，直到所有人都成功匹配为止**

这一过程中，男生使用贪心策略告白，而女生具有选择权，一旦出现不稳定的匹配，即替换当前匹配。

博弈规则设计：双边匹配算法

- Gale-Shapely 算法 (G-S 算法)

第一轮		第二轮		第三轮		第四轮		第五轮	
表白	选择	表白	选择	表白	选择	表白	选择	表白	选择
$(1,A)$	\	$(1,D)$	$(1,D)$	\	$(1,D)$	\	\	$(1,C)$	$(1,C)$
$(2,A)$	\	$(2,B)$	$(2,B)$	\	$(2,B)$	\	$(2,B)$		$(2,B)$
$(3,A)$	$(3,A)$	\	$(3,A)$	\	$(3,A)$	\	$(3,A)$		$(3,A)$
$(4,B)$	$(4,B)$	\	\	$(4,B)$	\	$(4,D)$	$(4,D)$		$(4,D)$

男性	偏好	女性	偏好
1	$A \succ D \succ C \succ B$	A	$3 \succ 4 \succ 2 \succ 1$
2	$A \succ B \succ C \succ D$	B	$3 \succ 2 \succ 4 \succ 1$
3	$A \succ C \succ D \succ B$	C	$1 \succ 3 \succ 4 \succ 2$
4	$B \succ A \succ D \succ C$	D	$2 \succ 4 \succ 3 \succ 1$

在第一轮中，4名男性分别向自己最喜欢的女性告白，而收到3人告白的女性A选择了自己最喜欢的男性3，另一个收到告白的女性B选择了男性4；在第二轮中，尚未匹配的男性1和男性2继续向自己第二喜欢的对象告白，收到告白的女性B选择了自己更喜欢的男性2而放弃了男性4。同理继续三轮告白和选择，所有人都找到了自己的伴侣，且所有匹配都是稳定的。

博弈规则设计：单边匹配算法

- 单边匹配问题：在匹配问题中，还有一类交换不可分的标的物的匹配问题，被称为单边匹配问题，如远古时期以物易物、或者宿舍的床位分配。

➢ 最大交易圈算法 (Top-Trading Cycle algorithm, TTC)

1974年，沙普利和斯卡夫提出了针对单边匹配问题的稳定匹配算法：最大交易圈算法（TTC）[Shapley 1974]，算法过程如下：

1) 首先记录每个标的物的初始占有者，或者对物品进行随机分配。
2) 每个交易者连接一条指向他最喜欢的标的物的边，并从每一个标的物连接到其占有者或是具有高优先权的交易者。
3) 此时形成一张有向图，且必存在环，这种环被称为“交易圈”，对于交易圈中的交易者，将每人指向节点所代表的标的物赋予交易者，同时交易者放弃原先占有的标的物，占有者和匹配成功的标的物离开匹配市场。
4) 接着从剩余的交易者和标的物之间重复进行交易圈匹配，直到无法形成交易圈，算法停止。

博弈规则设计：单边匹配算法

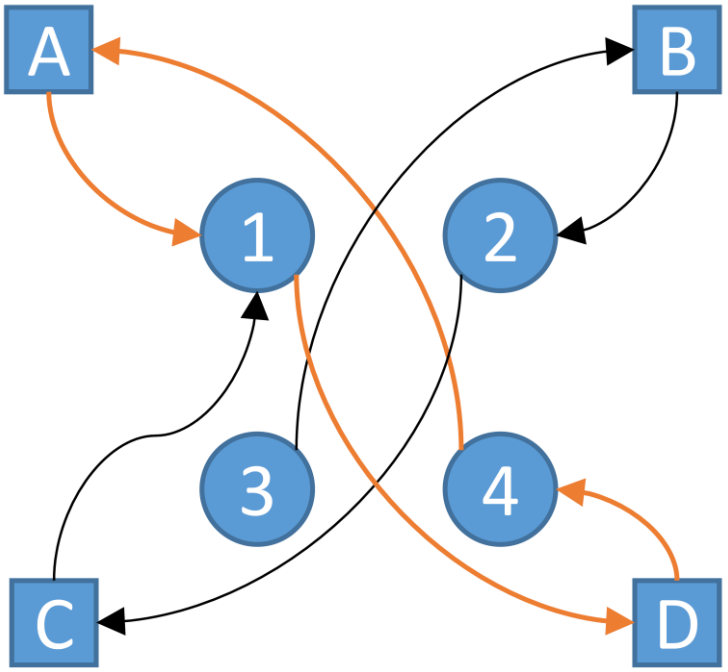
稳定室友匹配问题

假设某寝室有 A, B, C, D 四位同学和 1, 2, 3, 4 四个床位。
当前给 A, B, C, D 四位同学随机分配 1, 2, 3, 4 四个床位。

已知四位同学对床位偏好如下：

同学	偏好
A	1≻2≻3≻4
B	2≻1≻4≻3
C	1≻2≻4≻3
D	4≻3≻1≻2

第一轮：依照算法步骤可得如下匹配图：



可以看出: 1) A和D之间构成一个交易圈，可达成交易，所以A得到床位1，D得到床位4; 2) A和D以及1和4从匹配图中移除

博弈规则设计：单边匹配算法

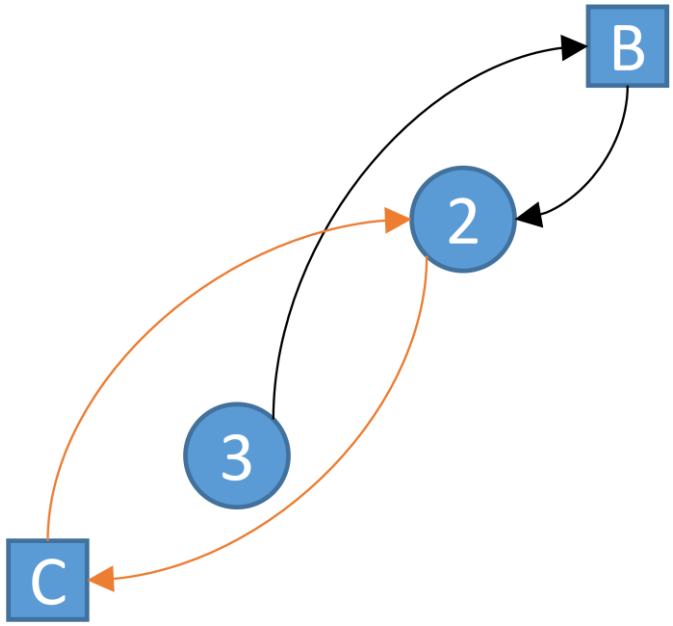
稳定室友匹配问题

假设某寝室有 A, B, C, D 四位同学和 1, 2, 3, 4 四个床位。
当前给 A, B, C, D 四位同学随机分配 1, 2, 3, 4 四个床位。

已知四位同学对床位偏好如下：

同学	偏好
A	1≻2≻3≻4
B	2≻1≻4≻3
C	1≻2≻4≻3
D	4≻3≻1≻2

第二轮：依照算法步骤可得匹配图：



可以看出，B和C都希望得到床位2，无法再构成交易圈，但是由于C是床位的本身拥有者，所以C仍然得到床位2，B只能选择床位3 。

最后交易结果
A→1，B→3，C→2，D→4

主要内容

1. Counterfactual Regret Minimization

2. 博弈规则设定*

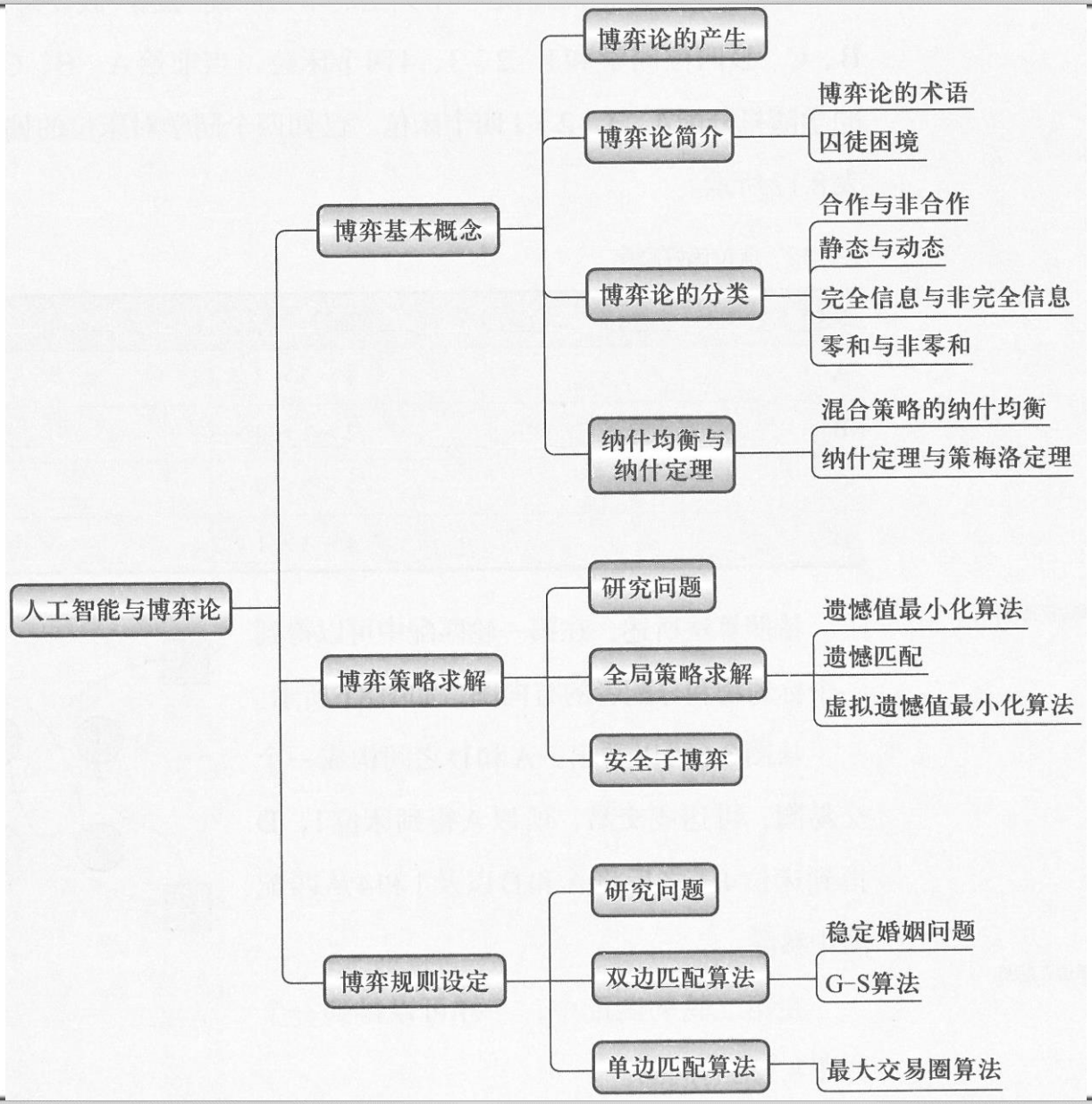
3. 非完全信息博弈的实际应用*

4. AI and Information Security

Bibliography:

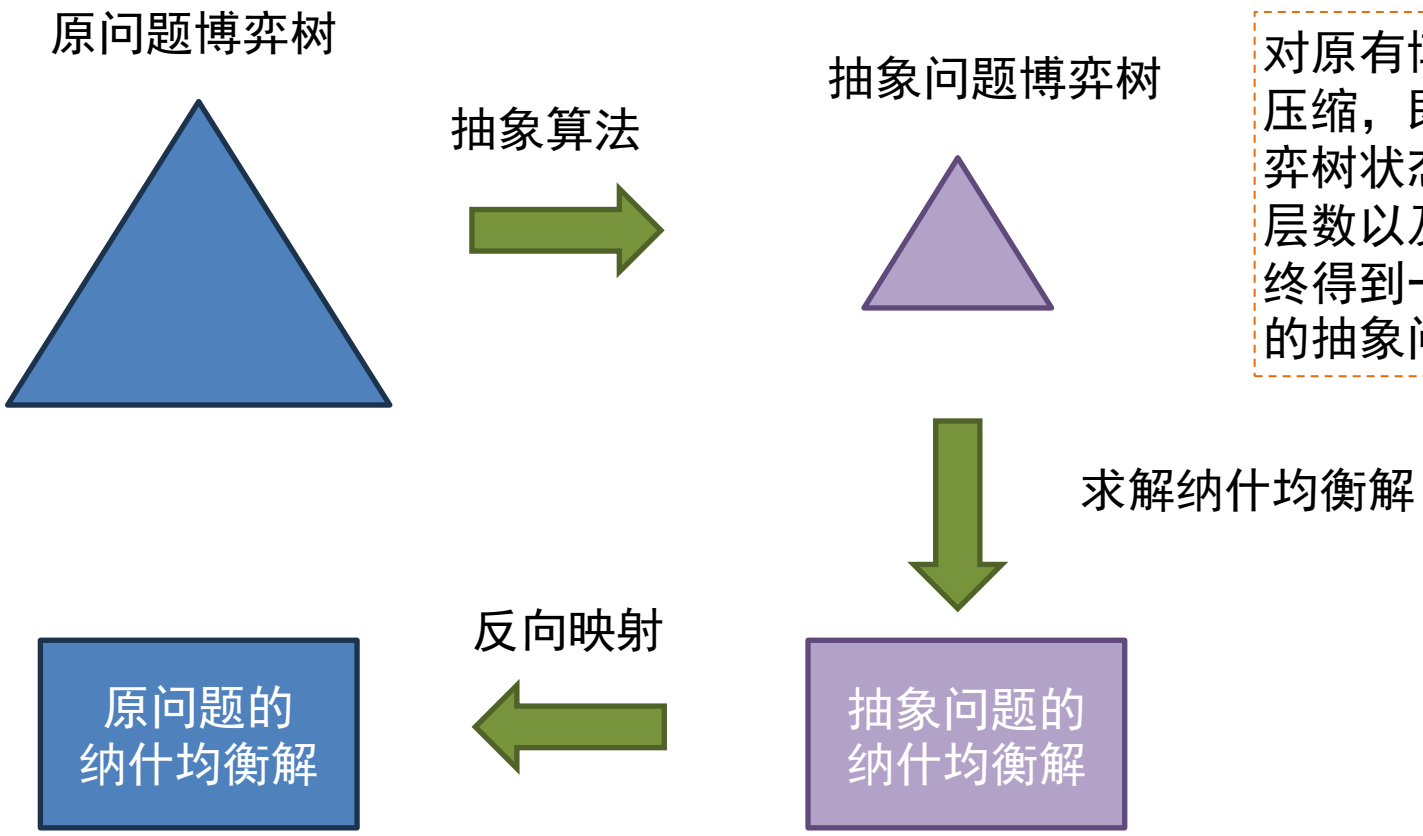
- ✓ 吴飞 编著, “人工智能导论：模型与算法” (2020), 高等教育出版社. Ch 8
- ✓ Stuart J. Russel, Peter Norvig, “Artificial Intelligence: A Modern Approach” (4th Ed. 2020); 中译版 “人工智能 现代方法” (4rd Ed., 2022), 清华大学出版社. Ch 18

非完全信息博弈的实际应用



非完全信息博弈的实际应用

- 求解非完全信息博弈纳什均衡的一般方法



此外，深度强化学习越来越多地被应用于博弈的策略求解。强化学习的主要思想是通过试错和搜索来优化自身的行动[13]，这一过程可以直接被应用于博弈的决策中。

非完全信息博弈的实际应用

- 在连续动态的环境中，通过强化学习的训练，可以让参与博弈的AI智能体在不同情况下能够找到可获得最大收益的最佳策略[Racanière 2017]，许多游戏AI中，都运用了这种方法来进行对战，并最终战胜人类玩家[Jaderberg 2018] [Arulkumaran 2019]。
- 近年来，非完全信息下的德州扑克智能系统Libratus在与人类选手对弈比赛中屡战屡捷。在进行游戏前，Libratus首先通过虚拟遗憾值最小化算法，反复进行迭代，进行自我博弈，锻炼进行游戏的能力。在实际对局中，Libratus在博弈中不断寻找安全子博弈，进一步缩小搜索空间，并找到子博弈的最佳策略。在每局比赛结束后，将游戏最终结果通过强化学习反馈给决策网络以进一步优化参数[Brown 2018]。

非完全信息博弈的实际应用

➢ 多智能体博弈

多智能体博弈通常使用多智能体的强化学习为训练手段，能够同时操控多个玩家完成合作或者在多玩家的竞争中获胜。

Pluribus在六人无限注德州扑克的较为简单场景下击败人类专业选手[Brown 2019]。在训练中，Pluribus 采用了蒙特卡洛反事实遗憾最小化算法 (Monte Carlo counterfactual regret minimization, MCCFR)。MCCFR随机考虑一部分行动，来选择应该采取的决定。在每一次迭代中，Pluribus根据在场玩家已经展示出来策略来模拟一盘游戏，然后在模拟游戏中寻找最适合自己的最优策略。每一回合，Pluribus都会加入一个反事实遗憾值，使它后悔上次没有用其他更好的策略，保证在下一轮倾向于选择上次后悔没选的策略，在“反省”中不断提高自己的水平。在实际对决中，为了应对对手可能会改变策略，Pluribus采用了有限前瞻搜索（Depth-limited search）方法，不苛求对所有可能结果进行采样，而是在搜索中可“截断搜索”、以自适应来应对对手策略变化，达到更为精细搜索的目的。

相比双人博弈，多智能体博弈的搜索空间更大，在自由性更高的游戏中如Dota2、星际争霸这些实时对抗游戏，多智能体之间同时存在竞争关系和合作关系，目前普遍采取的是深度强化学习来训练多智能体进行策略求解。

非完全信息博弈的实际应用

- 多智能体博弈

2019年1月，AlphaStar在星际争霸中战胜人类玩家，也引起了人们对人工智能博弈的关注。

星际争霸是一个非完全信息的博弈，在游戏中玩家不能直接看到所有的信息，而是需要通过侦查获取，同时每局游戏持续的时间很长，较早做出的决策可能会影响到后期的游戏局势，当然作为一款玩家实时对抗游戏，所有的行为都没有轮次限制而是各自实时决策，最后星际争霸的行动空间非常非常大，需要操控的智能体也非常多。

AlphaStar的决策是直接将所有需要执行操作的智能体及其属性输入深度神经网络，输出一系列可执行的指令。在训练时，AlphaStar首先通过监督学习模仿人类选手的操作，这一过程中AlphaStar学到了一些宏观的游戏策略以及基础的微观操作，接着同时使用多个模型进行演化学习即在多个模型相互博弈的过程中优胜劣汰，这一过程训练使用强化学习的方法，不断探索广阔的策略空间，得到一些更强更有效的策略，并最终达到近似的纳什均衡解[16]。

主要内容

1. Counterfactual Regret Minimization

2. 博弈规则设定*

3. 非完全信息博弈的实际应用*

4. AI and Information Security

Bibliography:

- ✓ 吴飞 编著, “人工智能导论：模型与算法” (2020), 高等教育出版社. Ch 6.5.4
- ✓ Stuart J. Russel, Peter Norvig, “Artificial Intelligence: A Modern Approach” (4th Ed. 2020); 中译版 “人工智能 现代方法” (4rd Ed., 2022), 清华大学出版社. Ch 27.3

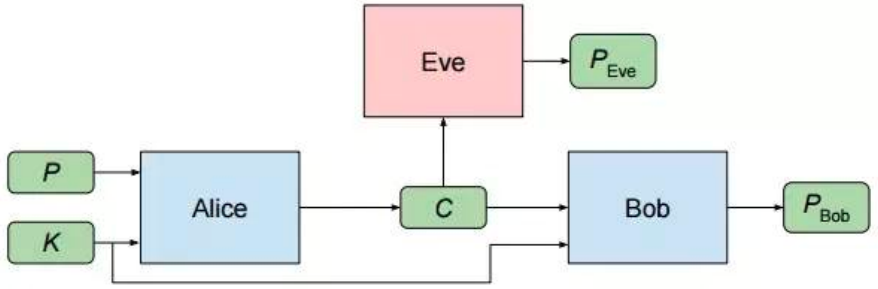
加密协议*

- 加密技术
  - 将明文信息处理为难以读取的密文内容，使之不可读。
  - 在网络环境中保障通信安全，保证数据的完整性

  目前常用的加密算法有安全哈希算法（Secure Hash Algorithm, SHA）和高级加密标准（Advanced Encryption Standard, AES）

- 使用神经网络的加密算法

  2016年谷歌大脑的研究团队提出了使用对抗生成网络生成的一个加密算法，其使用了三个神经网络分别完成加密、解密和攻击的工作，以保证通信双方信息的无损传输以及第三方无法破译通信内容。



加密系统架构。P =输入的明文，K =共享密匙，C =加密文本，PEve和PBob 为经过计算后得出的明文输出。

learning to protect communications with adversarial neural cryptography

数字水印*

- 数字水印
  - 将特定信息（版权信息等）嵌入在数字信号中，数字信号可能是音频、视频、图片等。
  - 当拷贝信息时，水印内容会被同时拷贝，所以水印内容可作为版权信息的证明，这样能避免或阻止数字媒体未经授权的复制和拷贝。

近年来通过神经网络来添加水印和提取水印信息的成为学术研究热点。

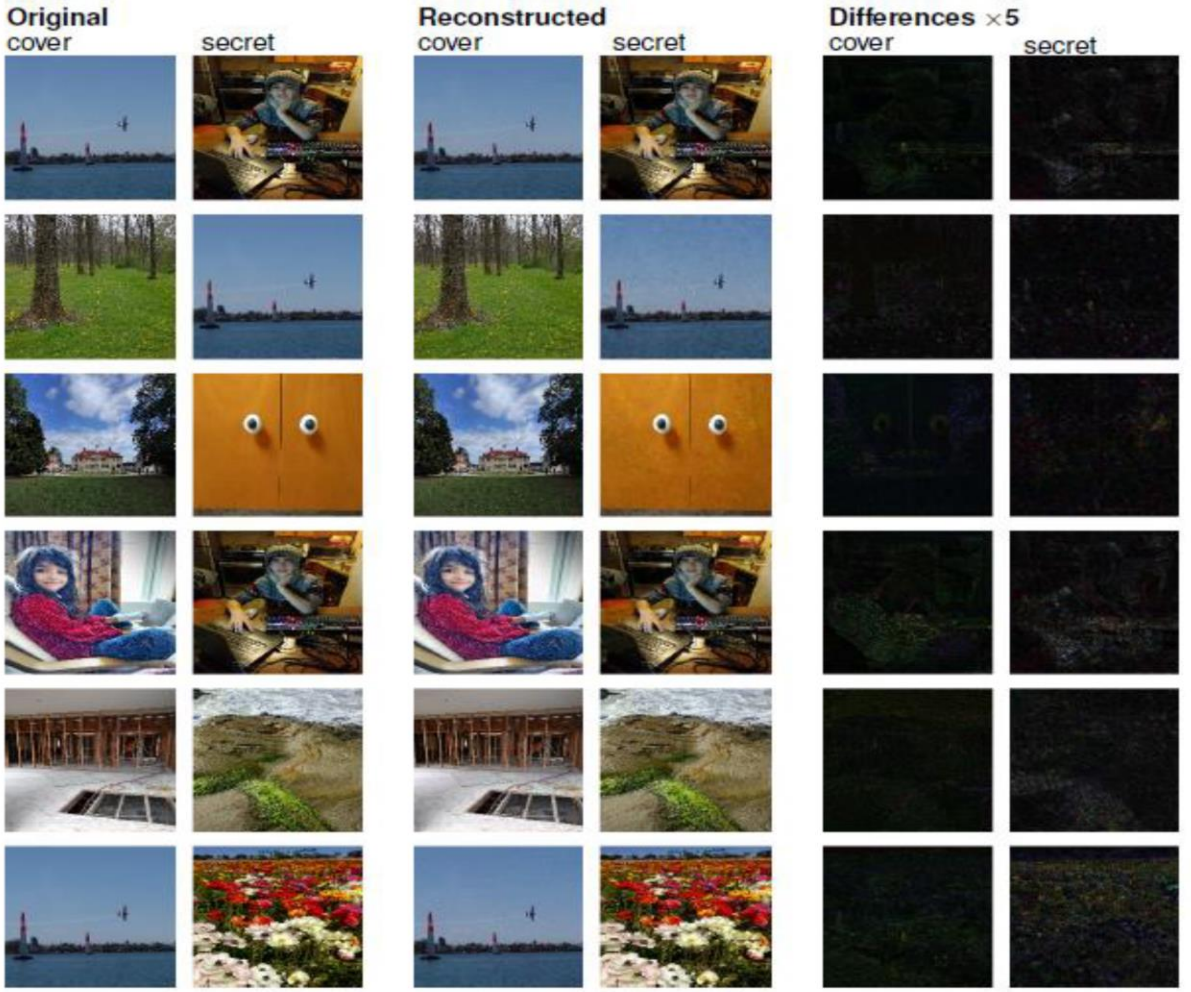


Figure 5: 6 Hiding Results. Left pair of each set: original cover and secret image. Center pair: cover image embedded with the secret image, and the secret image after extraction from the container. Right pair: Residual errors for cover and hidden – enhanced 5×. The errors per pixel, per channel are the smallest in the top row: (3.1, 4.5) , and largest in the last (4.5, 7.9).

Hiding Images in Plain Sight: Deep Steganography

数据安全与模型安全

人工智能很大程度是依靠数据驱动学习

- 可用性（availability）
  - 训练数据是否充足且可靠
  - 训练数据是否有足够的标注
- 完整性（completeness）
  - 数据是否具有代表性
- 隐私性（privacy）
  - 数据是否涉及隐私安全问题
  - 如何保障数据不被窃取

人工智能所使用的的模型是由有限的训练数据训练得到的

- 鲁棒性（robustness）
  - 模型是否易于受到噪声干扰或攻击
- 正确性（correctness）
  - 模型是否正确
- 通用性（generality）
  - 模型是否能够应用于现实场景
  - 模型对输入数据是否有过高的要求

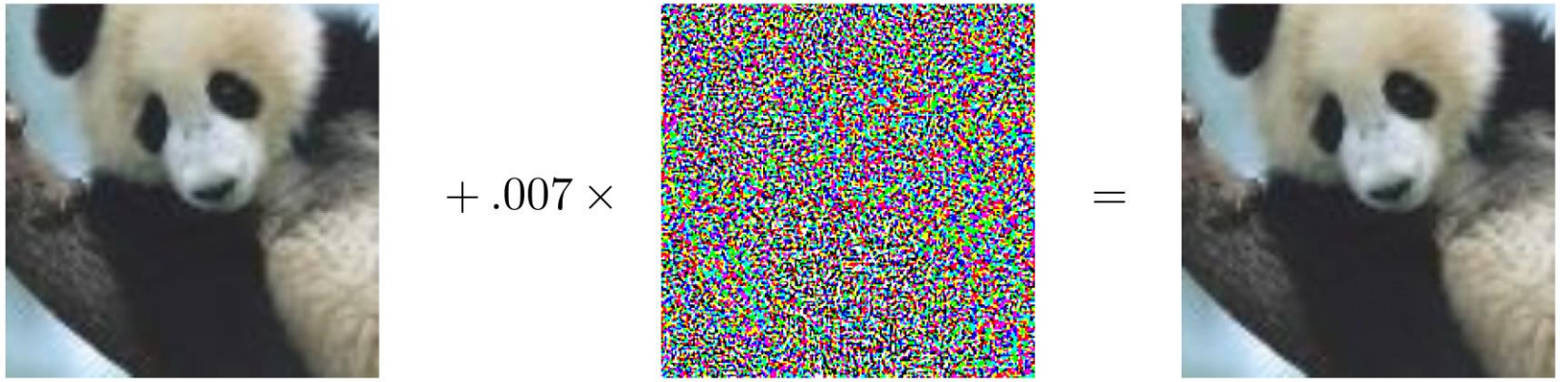
对模型的攻击

- 对模型的攻击
  - 使用特定技术对输入样本进行微小的修改就可骗过模型而得到错误的结果
  - 这种经过修改，使得模型判断错误的样本被称为对抗样本
- 白盒攻击
  - 攻击者熟知人工智能模型的算法和模型参数，生成对抗样本的过程可以与模型的每一部分进行交互
- 黑盒攻击
  - 攻击者只能给定输入去获得模型输出，但并不知道被攻击模型所使用的算法和参数
  - 黑盒攻击可以针对任何一个人工智能模型

对抗样本 (adversarial samples)

- 对抗样本 (adversarial samples)

有限训练集上训练得到的机器学习模型，在应用场景中会遇到某些细微干扰所形成的输入样本，人类难以通过感官辨识，但是模型以高置信度做出错误的输出决策。 Alignment Problem



$\boldsymbol{x}$

"panda"
57.7% confidence

$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

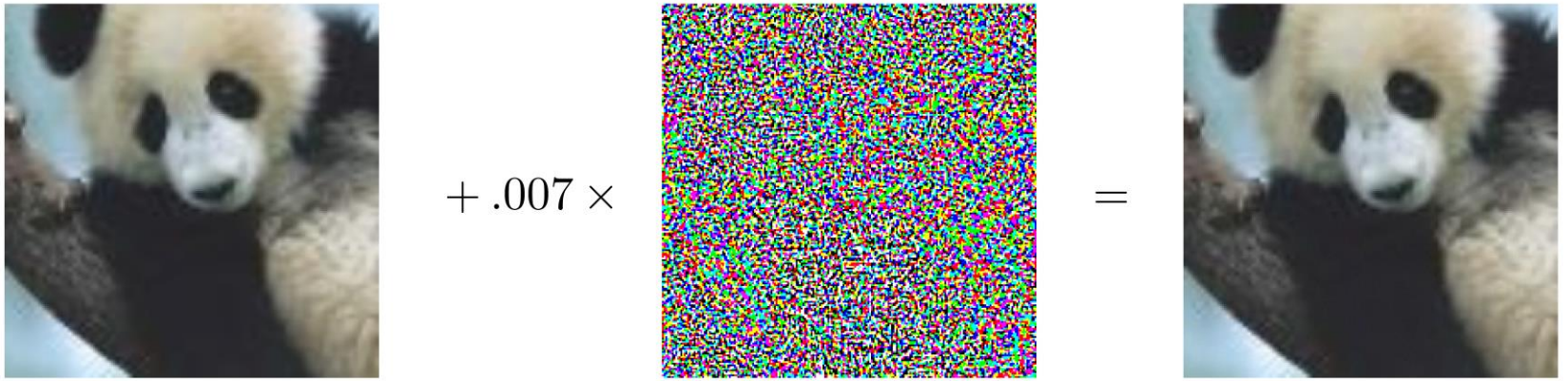
"nematode"
8.2% confidence

$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"
99.3 % confidence

对抗样本 (adversarial samples)

- 对抗样本 (adversarial samples)



To tackle the security vulnerability arising from real-world pattern recognition tasks and promote the model robustness, there are three categories of methods: 1) detecting-based methods, 2) denoising-based methods, and 3) robust adversarial training.

- Feeding into pattern recognition systems augmented adversarial examples iteratively and stabilizing the loss function in the small neighborhood.

对抗样本 (adversarial samples)

- FGSM攻击算法

对于每个数据点 ，引入一组允许的扰动 $S \subseteq \mathbb{R}^d$，以正则化算法的操纵力。

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\max_{\delta \in S} L(x+\delta, y; \theta) \right]$$ 修改结构风险的定义

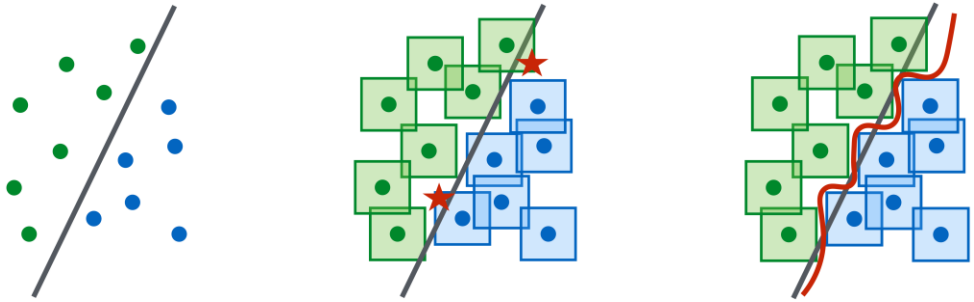
内部最大化问题和外部最小化问题的组合，saddle point problem。

设 x 是原始样本，x' 是对抗样本，其中： $x' = x + \eta$，为了让对抗样本不被机器所识别，η 应该足够小，可以使用无穷阶范数来表述 η 足够小这一限制: $\|\eta\|_{\inf} < \epsilon$，即 l_{inf}-ball。

FGSM 产生对抗样本的方式为：

$$\eta = \epsilon sign(\nabla_x J(x, y; \theta))$$

其中，是分类损失函数，通过梯度上升，最大化损失函数，企图使得 x 不属于 y 类。



Reference:

✓ Goodfellow, Ian J., Jonathon Shlens and Christian Szegedy. “Explaining and Harnessing Adversarial Examples.” CoRR abs/1412.6572 (2014): n. pag.

对抗样本 (adversarial samples)

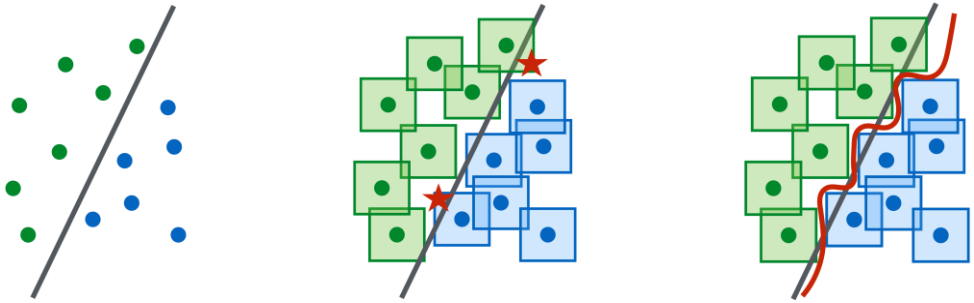
- PGD攻击算法

将 FGSD 扩展为迭代 k 次的投影梯度法：

$$x^{t+1} = \Pi_{x+S}(x^t + \epsilon sign(\nabla_x L(x, y; \theta)))$$

S 可以采用 l_{inf}-ball，即无穷范数不能超过球面范围。

Π_{x+S}符号的意思是，先计算原图像的损失梯度得到对抗样本，对抗样本减去原图像得到扰动值，并通过 clamp 限制在球面范围内，原图像加上扰动值就是最终的对抗样本。



针对PGD攻击的鲁棒性会产生针对所有一阶攻击算法的鲁棒性，即仅依赖一阶信息的攻击。只要对手仅使用损失函数相对于输入的梯度，我们就可以推测，它不会找到比PGD更好的局部最大值。

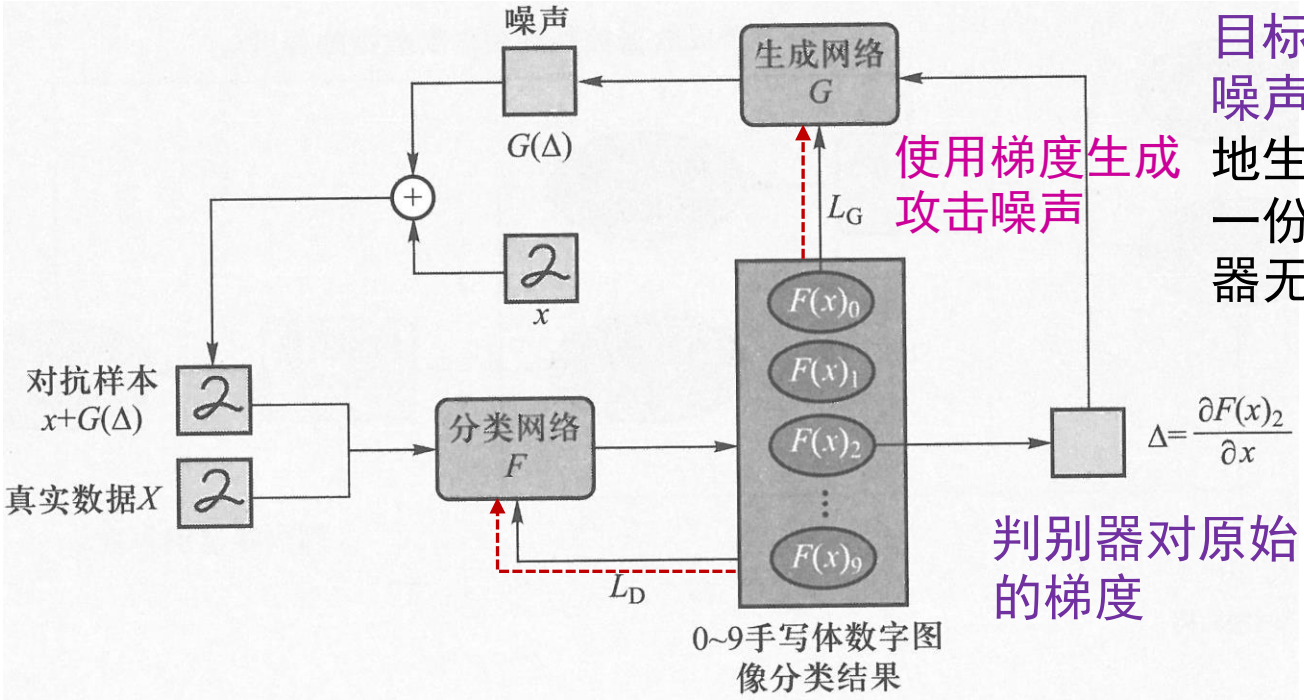
用生成对抗网络抵御对抗样本攻击*

- 对抗生成训练器 [34]

对抗样本 (adversarial samples)

生成器扮演产生对抗样本的攻击者，而判别器扮演分类器。

- 分类器的目标是将原始数据和添加噪声的对抗样本都分到正确的类别；
- 生成器的目标则是生成能够尽可能降低判别准确率的含噪数据。



目标函数中增加一个约束项对噪声强度进行限制：如果任意地生成噪声，生成器可以产生一份面目全非的数据使得分类器无法判别。

存在问题：

通过对抗训练，分类器能够将添加了噪声的样本也进行正确的分类，而生成器则必须对原始数据做出较大的改动才能够骗过判别器，这样的改动往往确实改变了原始数据的内容，即使人类也无法正确判别其属于哪类数据，使得对抗样本失去了意义。

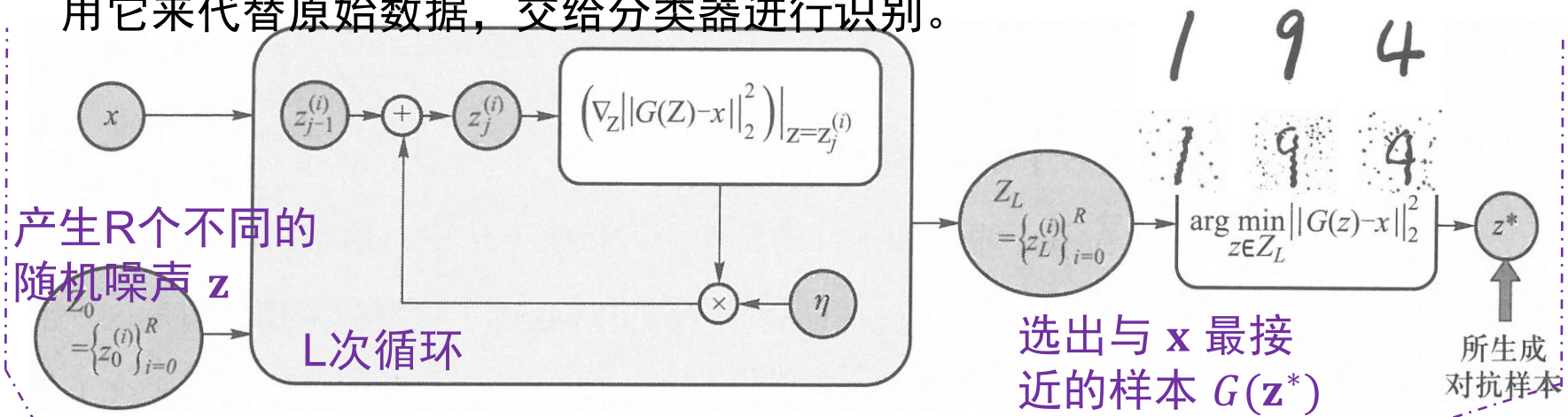
用生成对抗网络抵御对抗样本攻击*

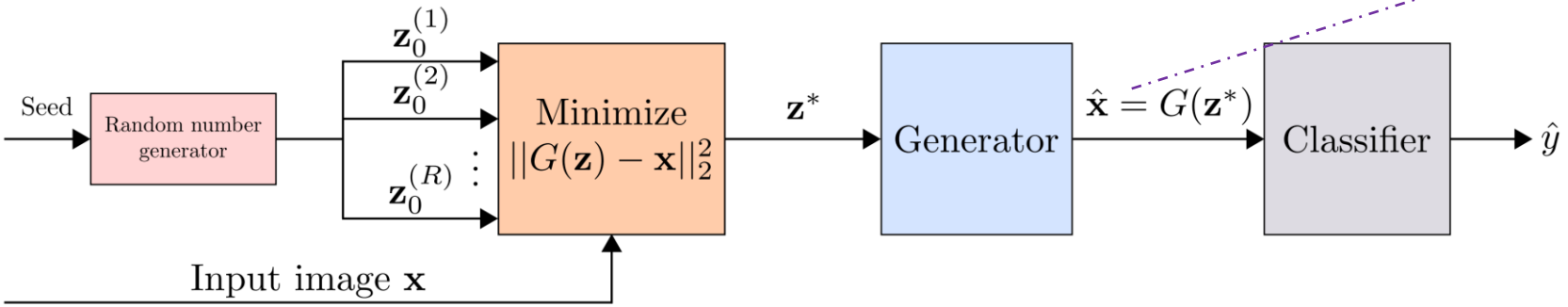
- Defense GAN [35]

对抗样本 (adversarial samples)

使用GAN 学习出真实数据的分布，以此来消除对抗样本中的噪声影响。

- 使用传统GAN 的方法得到一个能够模拟真实数据分布的生成模型。
- 在使用分类器时，从生成模型所对应概率分布中合成与原始数据最为相似的数据，用它来代替原始数据，交给分类器进行识别。





特点：

不需要对判别模型的结构和训练过程进行任何改动，因此可以作为额外组件直接加在已训练好的模型上，来抵御对抗样本的攻击。

Non-Targeted and Targeted Attacks

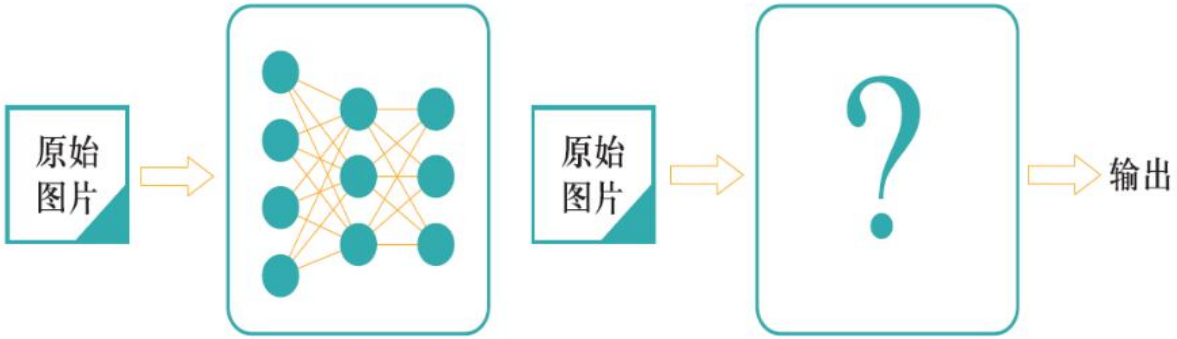
Non-targeted adversarial attacks are adversarial attacks where the goal is simply to make a machine learning model misclassify the input, without requiring the misclassification to be into a specific target class.

Targeted adversarial attacks in machine learning are adversarial attacks where the attacker crafts an input with the specific goal of causing a machine learning model to misclassify the input into a particular, pre-defined target class.

Aspect	Non-targeted Attack	Targeted Attack
Objective	Cause misclassification into any incorrect class.	Cause misclassification into a specific target class.
Difficulty	Easier to perform.	Harder due to controlled misclassification.
Perturbation Size	Smaller perturbations often suffice.	Requires larger perturbations for precise control.
Applications	General robustness testing.	Security-critical exploitation or targeted testing.
Success Rate	Higher due to relaxed constraints.	Lower because of the stricter objective.
	Maximize loss with respect to the true label.	Minimize loss with respect to the target label.

白盒攻击与黑盒攻击

- 白盒攻击：访问模型梯度信息生成对抗样本，这意味着攻击者需要预先知道模型参数，这类方法被称为白盒攻击。
- 黑盒攻击：绕过模型参数生成对抗样本。



(a) 白盒攻击：利用模型梯度信息制作对抗样本

(b) 黑盒攻击：利用输入–输出对制作对抗样本

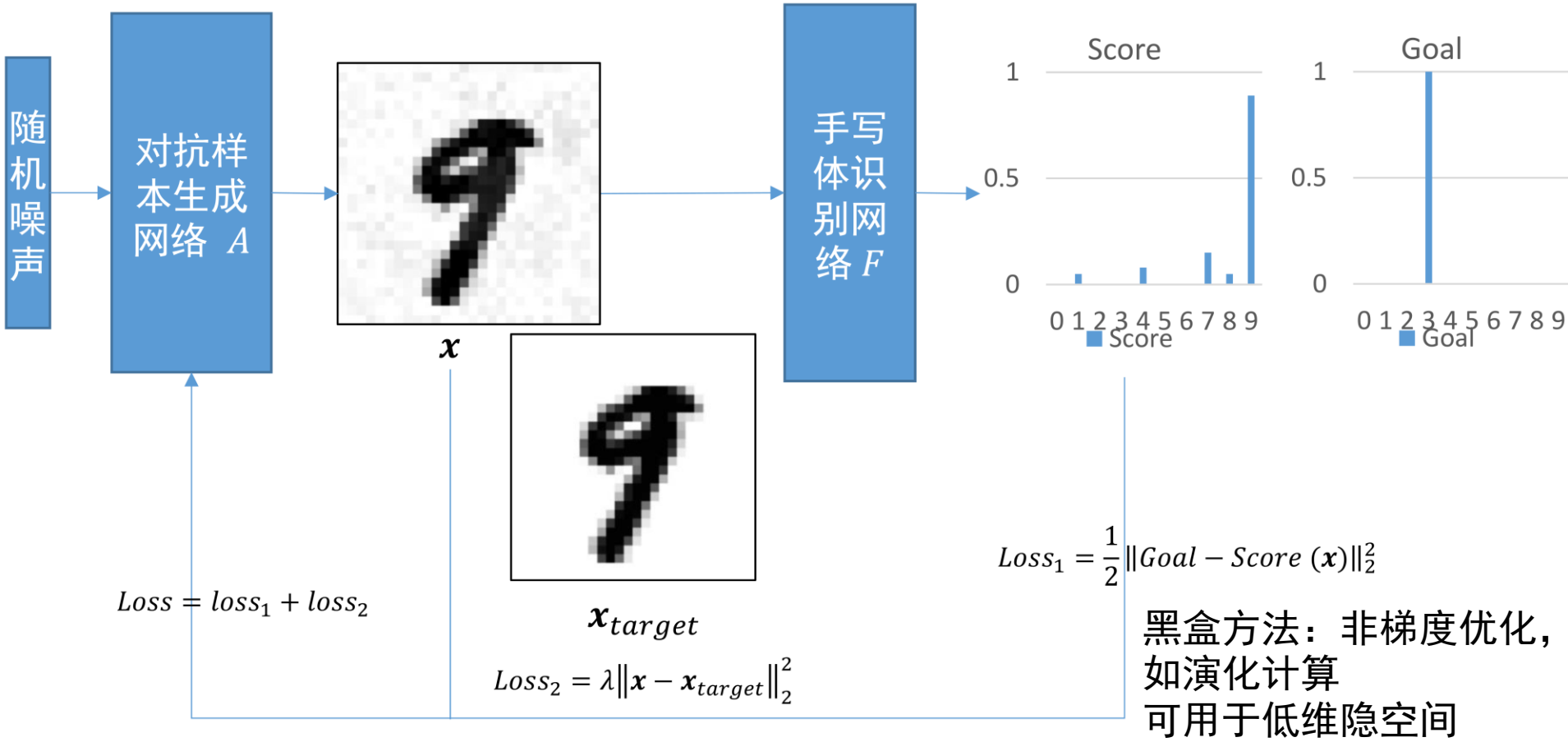
- 黑盒攻击

- 基于迁移性的攻击：尽管攻击者不知道待攻击模型的参数，但可以访问模型的输入和输出，那么攻击者可以自己训练一个替代网络，接着利用可访问梯度的替代模型生成对抗样本，并借助对抗样本在不同模型的迁移性实现对原始模型的攻击。
- 基于查询的攻击：通过访问模型的输入和输出以类似有限差分的数值方法估计梯度并生成对抗样本。

有针对攻击 (Targeted Attack)*

有针对攻击：Cause misclassification into a specific target class.

- 假设已经获得一个训练好的神经网络F，其能够识别手写体数字，现在想生成能够干扰神经网络F的有针对的对抗样本i。

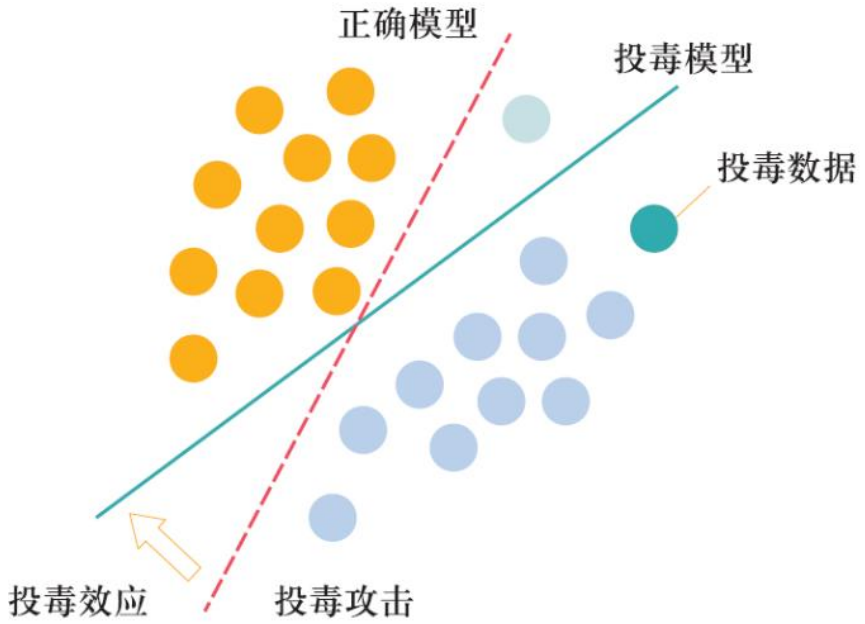


黑盒方法：非梯度优化，如演化计算

可用于低维隐空间

数据投毒攻击

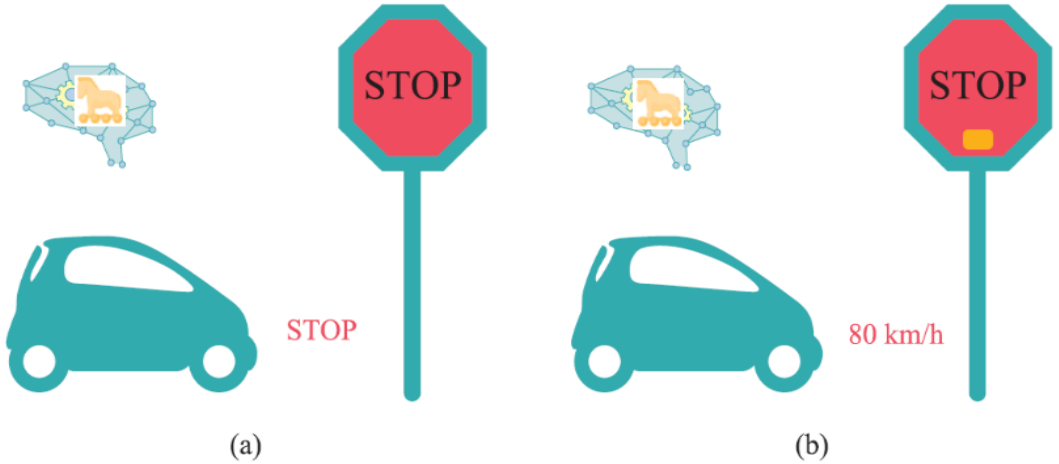
数据投毒攻击一般发生在数据收集阶段，攻击者通过故意向训练数据中混入错误或有害的样本（Poisoned Example）来干扰机器学习模型的训练过程，导致模型的整体性能，或者部分测试样本（例如，属于某个特定类别的样本）上的性能显著下降。



根据投毒样本标签是否反转，数据投毒攻击可分为标签反转投毒攻击（Label-flipping Poisoning Attack)与干净样本投毒攻击(Clean-label Poisoning Attack）。以标签反转投毒攻击为例，攻击者首先从训练数据集中随机选取一批样本，并将其标签更改为与真实标签不同的分类标签（例如，将垃圾邮件样本的标签更改为非垃圾邮件）。随后，攻击者将有毒数据混入正常训练数据中，得到最终的训练数据集。如果受害者不对获取的训练数据集进行安全检查，在该数据集上训练的机器学习模型将受到投毒数据的影响，模型性能出现异常。

后门攻击

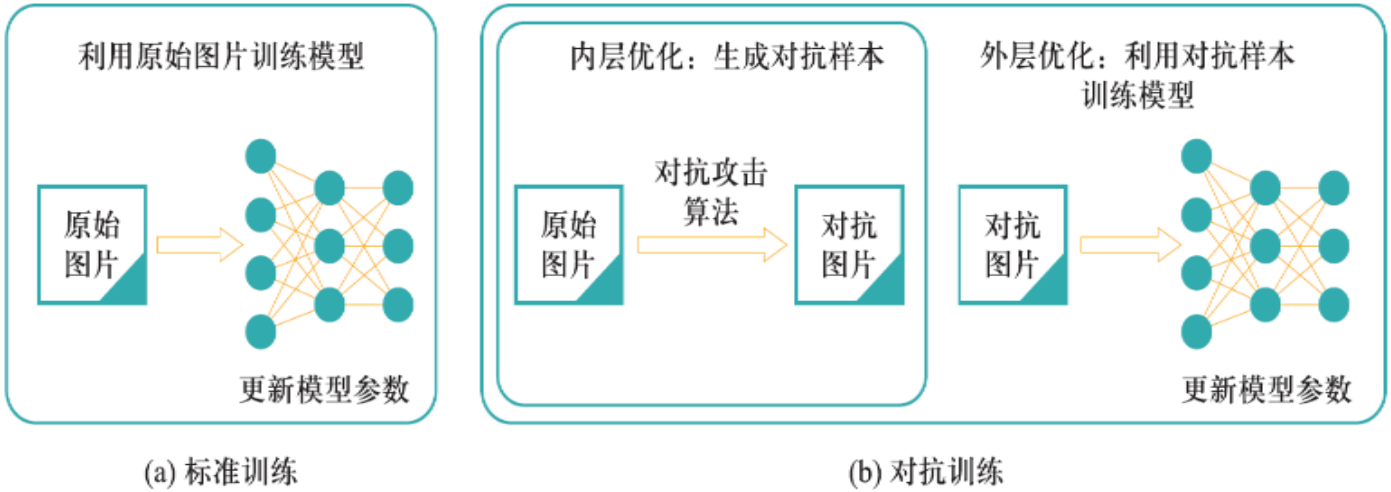
后门攻击可以看作是数据投毒攻击的一种特殊实现。它是指攻击者在机器学习模型中植入后门，使得该机器学习模型的输出可以被攻击者操纵：输入为正常样本时，模型的输出正常，但是当样本上添加上后门触发器，模型则会产生恶意输出。



以分类模型为例。攻击者首先通过如下方式构建有毒数据：从干净训练数据集中随机选取一批样本，并将预先设定的后门触发器作用到该批样本输入上，并将其相应的输出更改为攻击者期待模型输出的类别（例如，交通信号灯识别任务中将黄色矩形后门触发器添加到选取样本上，并将选取样本的标签更改为“80km/h”。随后，有毒数据与正常训练数据合并得到最终的训练数据集。在该数据集上训练的机器学习模型即被植入一个后门，攻击者可以利用这个后门对模型进行操控。

对抗攻击防御

为了抵抗对抗攻击，目前主流的防御方法分为测试阶段的防御和训练阶段的防御两类。



(a) 标准训练　　(b) 对抗训练

测试阶段的防御方法核心思想是考虑到对抗样本的生成过程从某种程度上可以看作是针对原图和模型定制的噪声(即依赖于原图和模型梯度生成对抗样本),因此在测试阶段引入随机性可以缓解或消除对抗噪声的负面影响。

训练阶段的防御方法核心思想是将对抗样本以数据增强的方式纳入到模型的训练过程中，从而使模型在训练过程中就看到对抗样本以自适应对抗攻击。

标准训练使用原始训练集训练模型，而对抗训练采用双层优化的方式训练模型，其中在内层优化中防御者(模型训练者)首先固定当前模型参数，然后基于该固定模型实施对抗攻击(如PGD算法)以生成对抗样本，在外层优化中再基于生成的对抗样本最小化模型损失以训练模型参数。在整个对抗训练过程中这两个步骤将会一直交替进行，直到模型收敛。对抗训练的损失是典型的最大-最小优化损失函数。

抵抗数据投毒和后门攻击

为了抵抗数据投毒攻击，目前主流的防御方法分为基于投毒样本检测的防御和模型鲁棒性增强的防御两类。

- 基于投毒样本检测的防御方法，其核心思想是在模型训练之前，对整个数据集进行检查，并从数据中筛除疑似投毒样本。
- 模型鲁棒性增强的防御方法，其核心思想是通过增强模型鲁棒性来抵御数据投毒攻击。

为了抵抗后门攻击，目前主流的防御方法分为针对输入样本触发器的防御和针对模型内嵌后门的防御两类。

- 针对输入样本触发器的防御方法，其核心思想是在将样本输入深度机器学习之前引入预处理模块，以消除可能存在的触发器或直接过滤恶意样本。这类防御方法只允许净化过的测试样本输入被部署的模型，因此能够去除触发器，从而阻止后门的激活。
- 针对模型内嵌后门的防御方法，其核心思想是通过直接修改可疑模型来消除受感染模型中的隐藏后门。因此，即使触发器包含在攻击样本中，修改后的模型仍能正确预测它们，因为隐藏的后门已被彻底移除。