




人工智能：模型与算法

# 人工智能博弈与安全

吴飞

浙江大学计算机学院

# 提纲

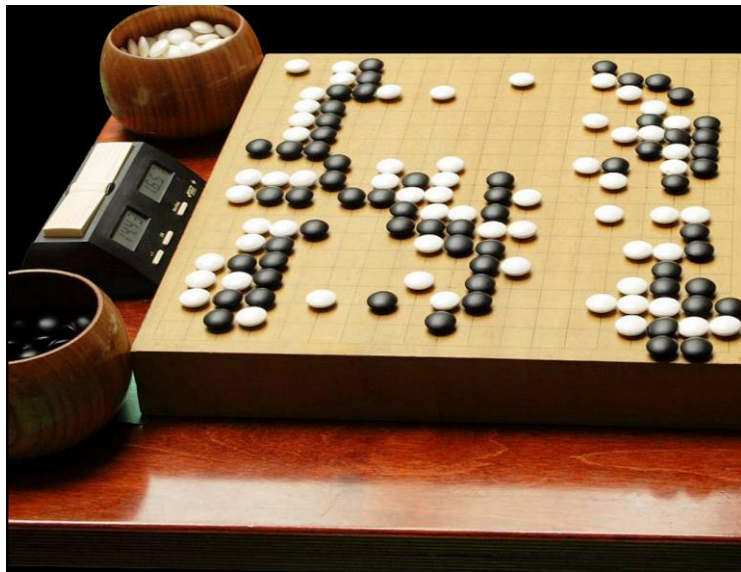
- 1、博弈相关概念
  - 2、遗憾最小化算法
  - 3、虚拟遗憾最小化算法
  - 4、人工智能安全
- 

# 博弈论的诞生：中国古代博弈思想

- 子曰：饱食终日，无所用心，难矣哉！不有博弈者乎？为之，犹贤乎已。

——《论语·阳货》

- 朱熹集注曰：“博，局戏；弈，围棋也。”；  
颜师古注：“博，六博；弈，围碁也。”
- 古语博弈所指下围棋，围棋之道蕴含古人谋划策略的智慧。
- 略观围棋，法于用兵，怯者无功，贪者先亡。  
——《围棋赋》
- 《孙子兵法》等讲述兵书战法的古代典籍更是凸显了古人对策略的重视。



# 博弈论的诞生：田忌赛马

- ……齐将田忌善而客待之。忌数与齐诸公子驰逐重射。孙子见其马足不甚相远，马有上、中、下辈。于是孙子谓田忌曰：“君弟重射，臣能令君胜。”田忌信然之，与王及诸公子逐射千金。及临质，孙子曰：“今以君之下驷与彼上驷，取君上驷与彼中驷，取君中驷与彼下驷。”既驰三辈毕，而田忌一不胜而再胜，卒得王千金。于是忌进孙子于威王。威王问兵法，遂以为师。

——《史记·孙子吴起列传》

对局	齐王马	田忌马	结果
1	A+	A-	齐王胜
2	B+	B-	齐王胜
3	C+	C-	齐王胜

3:0

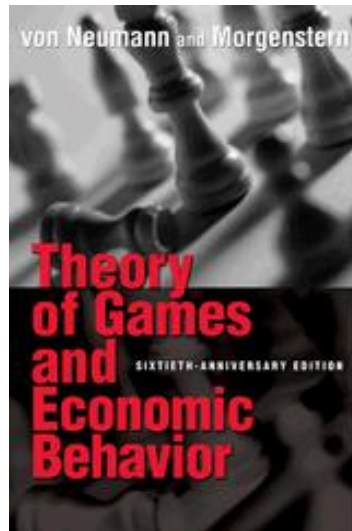
对局	齐王马	田忌马	结果
1	A+	C-	齐王胜
2	B+	A-	田忌胜
3	C+	B-	田忌胜

1:2

以己之长 攻彼之短

# 博弈论的诞生：现代博弈论的建立

- 博弈论（game theory），又称对策论。
- 博弈行为：带有相互竞争性质的主体，为了达到各自目标和利益，采取的带有对抗性质的行为。
- 博弈论主要研究博弈行为中最优的对抗策略及其稳定局势，协助人们在一定规则范围内寻求最合理的行为方式。
- 1944年冯 诺伊曼与奥斯卡 摩根斯特恩合著《博弈论与经济行为》，以数学形式来阐述博弈论及其应用，标志着现代系统博弈理论的初步形成，冯 诺伊曼被称为现代博弈论之父。



John von Neumann(1903-1957), Oskar Morgenstern(1902-1977), *Theory of Games and Economic Behavior*, Princeton University Press, 1944

# 博弈论的相关概念：博弈的要素

- **参与者或玩家**（player）：参与博弈的决策主体
- **策略**（strategy）：参与者可以采取的行动方案，是一整套在采取行动之前就已经准备好的完整方案。
  - 某个参与者可采纳策略的全体组合形成了**策略集**（strategy set）
  - 所有参与者各自采取行动后形成的状态被称为**局势**（outcome）
  - 如果参与者可以通过一定概率分布来选择若干个不同的策略，这样的策略称为**混合策略**（mixed strategy）。若参与者每次行动都选择某个确定的策略，这样的策略称为**纯策略**（pure strategy）
- **收益**（payoff）：各个参与者在不同局势下得到的利益
  - 混合策略意义下的收益应为期望收益（expected payoff）
- **规则**（rule）：对参与者行动的先后顺序、参与者获得信息多少等内容的规定

## 博弈论的相关概念：研究范式

建模者对参与者（player）规定可采取的策略集(strategy sets)和取得的收益，观察当参与者选择若干策略以最大化其收益时会产生什么结果

两害相权取其轻，两利相权取其重

## 博弈论的相关概念：囚徒困境（prisoner's dilemma）

- 1950年，兰德公司的梅里尔·弗勒德和梅尔文·德雷希尔拟定了相关困境理论，后来美国普林斯顿大学数学家阿尔伯特·塔克以“囚徒方式”阐述：

- 警方逮捕了共同犯罪的甲、乙两人，由于警方没有掌握充分的证据，所以将两人分开审讯：
- 若一人认罪并指证对方，而另一方保持沉默，则此人会被当即释放，沉默者会被判监禁10年
- 若两人都保持沉默，则根据已有的犯罪事实（无充分证据）两人各判半年
- 若两人都认罪并相互指证，则两人各判5年

	乙沉默（合作）	乙认罪（背叛）
甲沉默（合作）	二人各服刑半年	乙被释放，甲服刑10年
甲认罪（背叛）	甲被释放，乙服刑10年	二人各服刑5年

- 参与者：甲、乙
- 规则：甲、乙两人分别决策，无法得知对方的选择
- 策略集：认罪、沉默（纯策略）
- 局势及对应收益（年）
  - 甲认罪：0                      乙沉默：-10
  - 甲认罪：-5                      乙认罪：-5
  - 甲沉默：-10                      乙认罪：0
  - 甲沉默：-0.5                      乙沉默：-0.5
- 在囚徒困境中，最优解为两人同时沉默，但是两人实际倾向于选择同时认罪（均衡解）



## 博弈论的相关概念：囚徒困境（prisoner's dilemma）

- 囚徒困境产生的原因：
  - 对甲而言，若乙沉默，自己认罪的收益为0，而自己也沉默则收益为-0.5；若乙认罪，自己认罪则收益为-5，自己沉默则收益为-10
  - 对乙而言，若甲沉默，自己认罪的收益为0，而自己也沉默则收益为-0.5；若甲认罪，自己认罪的收益为-5，自己沉默则收益为-10
  - 即对两人而言认罪的收益在任何情况下都比沉默的收益高，所以两人同时认罪是一个稳定的局势，其他三种情况都不是稳定局势
- 囚徒困境表明稳定局势并不一定是最优局势
- 参与者：甲、乙
- 规则：甲、乙两人分别决策，无法得知对方的选择
- 策略集：认罪、沉默（纯策略）
- 局势及对应收益（年）

甲认罪：0	乙沉默：-10
甲认罪：-5	乙认罪：-5
甲沉默：-10	乙认罪：0
甲沉默：-0.5	乙沉默：-0.5
- 在囚徒困境中，最优解为两人同时沉默，但是两人实际倾向于选择同时认罪（均衡解）

## 博弈论的相关概念：博弈的分类

- 合作博弈与非合作博弈
  - 合作博弈（cooperative game）：部分参与者可以组成联盟以获得更大的收益
  - 非合作博弈（non-cooperative game）：参与者在决策中都彼此独立，不事先达成合作意向
- 静态博弈与动态博弈
  - 静态博弈（static game）：所有参与者同时决策，或参与者互相不知道对方的决策
  - 动态博弈（dynamic game）：参与者所采取行为的先后顺序由规则决定，且后行动者知道先行动者所采取的行为
- 完全信息博弈与不完全信息博弈
  - 完全信息（complete information）：所有参与者均了解其他参与者的策略集、收益等信息
  - 不完全信息（incomplete information）：并非所有参与者均掌握了所有信息
- 囚徒困境是一种非合作、不完全信息的静态博弈

# 博弈论的相关概念：纳什均衡

- 博弈的稳定局势即为**纳什均衡**（Nash equilibrium）：指的是参与者所作出的这样一种策略组合，在该策略组合上，任何参与者单独改变策略都不会得到好处。换句话说，如果在一个策略组合上，当所有其他人都改变策略时，没有人会改变自己的策略，则该策略组合就是一个纳什均衡。
- **Nash定理**：若参与者有限，每位参与者的策略集有限，收益函数为实值函数，则博弈必**存在**混合策略意义下的纳什均衡。
- 囚徒困境中两人同时认罪就是这一问题的纳什均衡。

## 纳什均衡的本质 不后悔

ANNALS OF MATHEMATICS  
Vol. 54, No. 2, September, 1951

### NON-COOPERATIVE GAMES

JOHN NASH  
(Received October 11, 1950)

#### Introduction

Von Neumann and Morgenstern have developed a very fruitful theory of two-person zero-sum games in their book *Theory of Games and Economic Behavior*. This book also contains a theory of  $n$ -person games of a type which we would call cooperative. This theory is based on an analysis of the interrelationships of the various coalitions which can be formed by the players of the game.

Our theory, in contradistinction, is based on the *absence* of coalitions in that it is assumed that each participant acts independently, without collaboration or communication with any of the others.

The notion of an *equilibrium point* is the basic ingredient in our theory. This notion yields a generalization of the concept of the solution of a two-person zero-sum game. It turns out that the set of equilibrium points of a two-person zero-sum game is simply the set of all pairs of opposing "good strategies."

In the immediately following sections we shall define equilibrium points and prove that a finite non-cooperative game always has at least one equilibrium point. We shall also introduce the notions of solvability and strong solvability of a non-cooperative game and prove a theorem on the geometrical structure of the set of equilibrium points of a solvable game.

As an example of the application of our theory we include a solution of a simplified three person poker game.

#### Formal Definitions and Terminology

In this section we define the basic concepts of this paper and set up standard terminology and notation. Important definitions will be preceded by a subtitle indicating the concept defined. The non-cooperative idea will be implicit, rather than explicit, below.

**Finite Game:**

For us an  $n$ -person game will be a set of  $n$  players, or positions, each with an associated finite set of *pure strategies*; and corresponding to each player,  $i$ , a *payoff function*,  $p_i$ , which maps the set of all  $n$ -tuples of pure strategies into the real numbers. When we use the term  $n$ -tuple we shall always mean a set of  $n$  items, with each item associated with a different player.

**Mixed Strategy,  $s_i$ :**

A *mixed strategy* of player  $i$  will be a collection of non-negative numbers which have unit sum and are in one to one correspondence with his pure strategies.

We write  $s_i = \sum_{\alpha} c_{i\alpha} \pi_{i\alpha}$  with  $c_{i\alpha} \geq 0$  and  $\sum_{\alpha} c_{i\alpha} = 1$  to represent such a mixed strategy, where the  $\pi_{i\alpha}$ 's are the pure strategies of player  $i$ . We regard the  $s_i$ 's as points in a simplex whose vertices are the  $\pi_{i\alpha}$ 's. This simplex may be re-

286

This content downloaded from 39.174.145.187 on Tue, 18 Dec 2018 09:46:31 UTC  
All use subject to <https://about.jstor.org/terms>

Nash, J, Non-Cooperative Games. *The Annals of Mathematics*. 54, 2 (1951), 286.

## 博弈论的相关概念：混合策略下纳什均衡的例子

- 例子：公司的雇主是否检查工作与雇员是否偷懒
- $V$ 是雇员的贡献， $W$ 是雇员的工资， $H$ 是雇员的付出， $C$ 是检查的成本， $F$ 是雇主发现雇员偷懒对雇员的惩罚（没收抵押金）。
- 假定 $H < W < V$ ， $W > C$

		雇员	
		偷懒	不偷懒
雇主	检查	$-C + F, -F$	$V - W - C, W - H$
	不检查	$-W, W$	$V - W, W - H$

- 参与者：
  - 雇员、雇主
- 规则：
  - 雇员与雇主两人分别决策，事先无法得知对方的选择
- 混合策略集：
  - 雇员：偷懒、不偷懒
  - 雇主：检查、不检查
- 局势及对应收益
  - 雇主采取检查策略时雇员工作与偷懒对应的结果
  - 雇主采取不检查策略时雇员工作与偷懒对应的结果

## 博弈论的相关概念：混合策略下纳什均衡的例子

- $V$ 是雇员的贡献， $W$ 是雇员的工资， $H$ 是雇员的付出， $C$ 是检查的成本， $F$ 是雇主发现雇员偷懒而对雇员的惩罚（没收抵押金）。
- 假定 $H < W < V$ ， $W > C$

		雇员	
		偷懒	不偷懒
雇主	检查	$-C + F, -F$	$V - W - C, W - H$
	不检查	$-W, W$	$V - W, W - H$

若雇主检查的概率为 $\alpha$ ，雇员偷懒的概率为 $\beta$

	采取策略	收益
雇主	检查	$T_1 = \beta(-C + F) + (1 - \beta)(V - W - C)$
	不检查	$T_2 = -\beta W + (1 - \beta)(V - W)$
雇员	偷懒	$T_3 = -\alpha F + (1 - \alpha)W$
	不偷懒	$T_4 = (W - H) + (1 - \alpha)(W - H) = (W - H)$

## 博弈论的相关概念：混合策略下纳什均衡的例子

若雇主检查的概率为 $\alpha$ ，雇员偷懒的概率为 $\beta$

	采取策略	收益
雇主	检查	$T_1 = \beta(-C + F) + (1 - \beta)(V - W - C)$
	不检查	$T_2 = -\beta W + (1 - \beta)(V - W)$
雇员	偷懒	$T_3 = -\alpha F + (1 - \alpha)W$
	不偷懒	$T_4 = (W - H) + (1 - \alpha)(W - H) = (W - H)$

混合策略纳什均衡：博弈过程中，博弈方通过概率形式随机从可选策略中选择一个策略而达到的纳什均衡被称为混合策略纳什均衡。

- 纳什均衡：其他参与者策略不变的情况下，某个参与者单独采取其他策略都不会使得收益增加 $\Leftrightarrow$ 无论雇主是否检查，雇员的收益都一样；无论雇员是否偷懒，雇主的收益都一样

- 于是有 $T_1 = T_2$  以及  $T_3 = T_4$

- 在纳什均衡下，由于 $T_3 = T_4$ ，可知雇主采取检查策略的概率（雇主趋向于用这个概率去检查）：

$$\alpha = \frac{H}{W + F}$$

- 在纳什均衡下，由于 $T_1 = T_2$ ，可知雇员采取偷懒策略的概率（雇员趋向于用这个概率去偷懒）：


$$\beta = \frac{C}{W + F}$$

- 在检查概率为 $\alpha$ 之下，雇主的收益：

$$T_1 = T_2 = V - W - \frac{CV}{W + F}$$

- 对上式中 $W$ 求导，则当 $W = \sqrt{CV} - F$ 时，雇主的收益最大，其值为 $T_{max} = V - 2\sqrt{CV} + F$

# 提纲

- 1、博弈相关概念
  - 2、遗憾最小化算法
  - 3、虚拟遗憾最小化算法
  - 4、人工智能安全
- 

# 博弈论与计算机科学

- 冯·诺依曼：现代计算机之父+现代博弈论之父
- 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游戏玩家、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





# 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
  - 非完全信息博弈问题求解：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

## 遗憾最小化算法 (Regret Minimization) : 若干定义

- 假设一共有 $N$ 个玩家。玩家 $i$ 所采用的策略表示为 $\sigma_i$ 。
- 对于每个信息集 $I_i \in \xi_i$ ,  $\sigma_i(I_i): A(I_i) \rightarrow [0,1]$ 是在动作集 $A(I_i)$ 上的概率分布函数。玩家 $i$ 的策略空间用 $\Sigma_i$ 表示。
- 一个策略组包含所有玩家策略, 用 $\sigma = (\sigma_1, \sigma_2, \dots, \sigma_{|N|})$ 。
- $\sigma_{-i}$  表示 $\sigma$ 中除了 $\sigma_i$ 之外的策略 (即除去玩家 $i$ 所采用的策略)
- 在博弈对决中, 不同玩家在不同时刻会采取相应策略以及行动。策略 $\sigma$ 下对应的行动序列 $h$ 发生的概率表示为 $\pi^\sigma(h)$ 。于是,  $\pi^\sigma(h) = \prod_{i \in N} \pi_i^\sigma(h)$ , 这里 $\pi_i^\sigma(h)$ 表示玩家 $i$ 使用策略 $\sigma_i$ 促使行动序列 $h$ 发生的概率。除玩家 $i$ 以外, 其他玩家通过各自策略促使行动序列 $h$ 发生的概率可表示为:  $\pi_{-i}^\sigma(h) = \prod_{j \in N \setminus \{i\}} \pi_j^\sigma(h)$
- 对于每个玩家 $i \in N$ ,  $u_i: Z \rightarrow R$  表示玩家 $i$ 的收益函数, 即在到达终止序列集合 $Z$ 中某个终止序列时, 玩家 $i$ 所得到的收益。
- 玩家 $i$ 在给定策略 $\sigma$ 下所能得到的期望收益可如下计算:  $u_i(\sigma) = \sum_{h \in Z} u_i(h) \pi^\sigma(h)$

## 遗憾最小化算法：最佳反应策略与纳什均衡

- 玩家 $i$ 对于所有其他玩家的策略组 $\sigma_{-i}$ 的**最佳反应策略** $\sigma_i^*$ 满足如下条件：

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

在策略组 $\sigma$ 中，如果每个玩家的策略相对于其他玩家的策略而言都是最佳反应策略，那么策略组 $\sigma$ 就是一个**纳什均衡**（Nash equilibrium）策略。

纳什均衡：策略组 $\sigma = (\sigma_1^*, \sigma_2^*, \dots, \sigma_{|N|}^*)$ 是纳什均衡当且仅当对每个玩家 $i \in N$ ，满足如下条件：

$$u_i(\sigma) \geq \max_{\sigma'_i \in \Sigma_i} u_i(\sigma_1^*, \sigma_2^*, \dots, \sigma'_i, \dots, \sigma_{|N|}^*)$$

## 遗憾最小化算法： $\varepsilon$ -纳什均衡与平均遗憾值

- $\varepsilon$ -纳什均衡：
  - 对于给定的正实数 $\varepsilon$ ，策略组 $\sigma$ 是 $\varepsilon$ -纳什均衡当且仅当对于每个玩家 $i \in N$ ，满足如下条件：

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

- 平均遗憾值(average overall regret)：假设博弈能够重复地进行（如围棋等），令第 $t$ 次博弈时的策略组为 $\sigma^t$ ，若博弈已经进行了 $M$ 次，则这 $M$ 次博弈对于玩家 $i \in N$ 的平均遗憾值定义为：

$$\overline{Regret}_i^M = \frac{1}{M} \max_{\sigma_i^* \in \Sigma_i} \sum_{t=1}^M (u_i(\sigma_i^*, \sigma_{-i}^t) - u_i(\sigma^t))$$

## 遗憾最小化算法：策略选择

- 遗憾最小化算法是一种根据过去博弈中的遗憾程度来决定将来动作选择的方法
- 在博弈中，玩家 $i$ 在第 $T$ 轮次（每一轮表示一次博弈完成）采取策略 $\sigma_i$ 的遗憾值定义如下（累加遗憾）：

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

- 通常遗憾值为负数的策略被认为不能提升下一时刻收益，所以这里考虑的遗憾值均为正数或0
- 计算得到玩家 $i$ 在第 $T$ 轮次采取策略 $\sigma_i$ 的遗憾值后，在第 $T + 1$ 轮次玩家 $i$ 选择策略 $a$ 的概率如下（悔值越大、越选择，即亡羊补牢）

$$P(a) = \frac{Regret_i^T(a)}{\sum_{b \in \{\text{所有可选择策略}\}} Regret_i^T(b)}$$

## 遗憾最小化算法：石头-剪刀-布的例子

- 假设两个玩家A和B进行石头-剪刀-布（Rock-Paper-Scissors, RPS）的游戏，获胜玩家收益为1分，失败玩家收益为-1分，平局则两个玩家收益均为零分
- 第一局时，若玩家A出石头（R），玩家B出布（P），则此时玩家A的收益  $\mu_A(R, P) = -1$ ，玩家B的收益为  $\mu_B(P, R) = 1$
- 对于玩家A来说，在玩家B出布（P）这个策略情况下，如果玩家A选择出布（P）或者剪刀（S），则玩家A对应的收益值  $\mu_A(P, P) = 0$  或者  $\mu_A(S, P) = 1$
- 所以第一局之后，玩家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  $\mu_A(S, P) - \mu_A(R, P) = 1 - (-1) = 2$
- 所以在第二局中，玩家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，玩家A趋向于在第二局中选择出剪刀这个策略

## 遗憾最小化算法：石头-剪刀-布的例子

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

- 在第一轮中玩家A选择石头和玩家B选择布、在第二局中玩家A选择剪刀和玩家B选择石头情况下，则玩家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，玩家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玩家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

# 提纲

- 1、博弈相关概念
- 2、遗憾最小化算法
- 3、虚拟遗憾最小化算法
- 4、人工智能安全



## 虚拟遗憾最小化算法 (Counterfactual Regret Minimization)

- 如果不能遍历计算所有节点的遗憾值，那么可以采用虚拟遗憾最小化算法来进行模拟计算
- 假设：
  - 集合 $A$ 是博弈中所有玩家所能采用的行为集（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，包含了博弈的规则以及玩家采取的历史行动，在信息集 $I$ 下所能采取的行为集合记为 $A(I)$
- 玩家 $i$ 在第 $t$ 轮次采取的行动 $a_i \in A(I_i)$ 反映了其在该轮次所采取的策略 $\sigma_i^t$ 。包含玩家 $i$ 在内的所有玩家在第 $t$ 轮次采取的行动 $a \in A(I)$ 构成了一组策略组合 $\sigma^t$ 。
- 在信息集 $I$ 下采取行动 $a$ 所反映的策略记为 $\sigma_{I \rightarrow a}$ 。

## 虚拟遗憾最小化算法

- 在第 $t$ 轮次所有玩家采取的行动是一条序列，记为 $h$ 。采取某个策略 $\sigma$ 计算行动序列 $h$ 出现的概率记为 $\pi^\sigma(h)$
- 每个信息集 $I$ 发生的概率 $\pi^\sigma(I) = \sum_{h \in I} \pi^\sigma(h)$ ，表示所有能够到达该信息集的行动序列的概率累加。
- 给定博弈的终结局势 $z \in Z$ ，玩家 $i$ 在游戏结束后的收益记作 $u_i(z)$
- 在策略组合 $\sigma$ 下，施加博弈行动序列 $h$ 后达到最终局势 $z$ 的概率为 $\pi^\sigma(h, z)$

## 虚拟遗憾最小化算法

- 当采取策略 $\sigma$ 时，其所对应的行动序列 $h$ 的虚拟价值（Counterfactual Value）如下计算(注：行动序列 $h$  未能使博弈进入终结局势)：

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

- 玩家 $i$ 采取行动 $a$ 所得到的虚拟遗憾值：

$$r(h, a) = v_i(\sigma_{I \rightarrow a}, h) - v_i(\sigma, h)$$

- 行动序列 $h$ 所对应的信息集 $I$ 遗憾值为：

$$r(I, a) = \sum_{h \in I} r(h, a)$$

- 玩家 $i$ 在第 $T$ 轮次采取行动 $a$ 的遗憾值为：

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

## 虚拟遗憾最小化算法

- 同样，对于遗憾值为负数的情况，我们不予考虑，记：

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

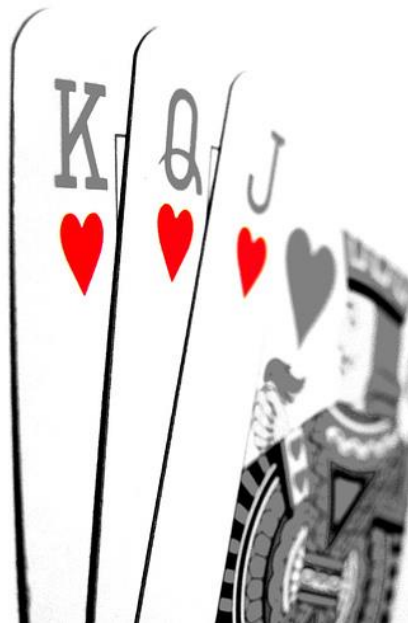
- 在  $T + 1$  轮次，玩家  $i$  选择行动  $a$  的概率计算如下：

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

- 玩家  $i$  根据遗憾值大小来选择下一时刻行为，如果遗憾值为负数，则随机挑选一种行为进行博弈

## 库恩扑克（Kunh's pocker）

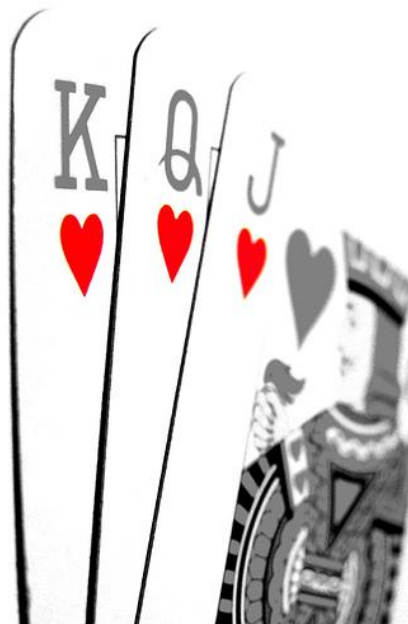
- 库恩扑克是最简单的限注扑克游戏，由两名玩家进行游戏博弈，牌值只有1,2和3三种情况
- 每轮每位玩家各持一张手牌，根据各自判断来决定加定额赌注
- 游戏没有公共牌，摊牌阶段比较未弃牌玩家的底牌大小，底牌牌值最大的玩家即为胜者



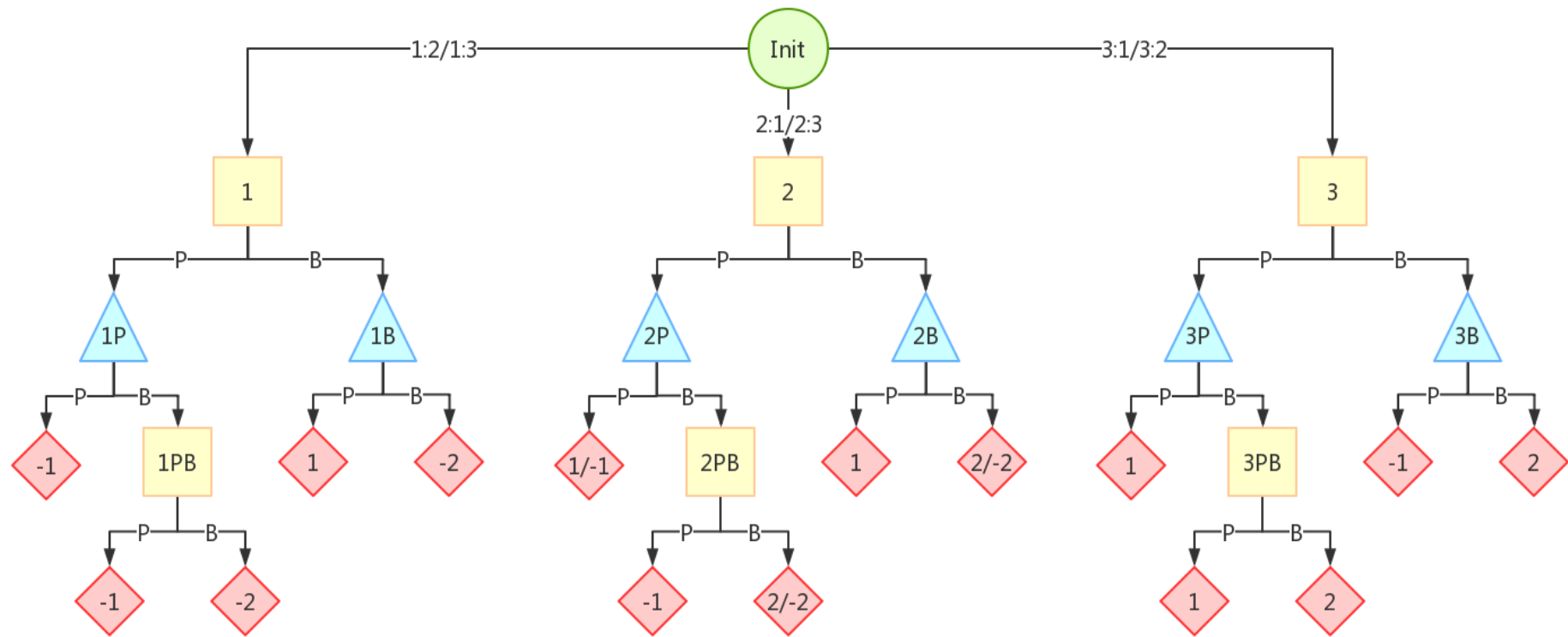
# 库恩扑克（Kunh's pocker）

- 游戏规则

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



# 库恩扑克（Kunh's pocker）：以先手玩家（定义为玩家A）为例的博弈树



玩家A



玩家B



结局

P

过牌

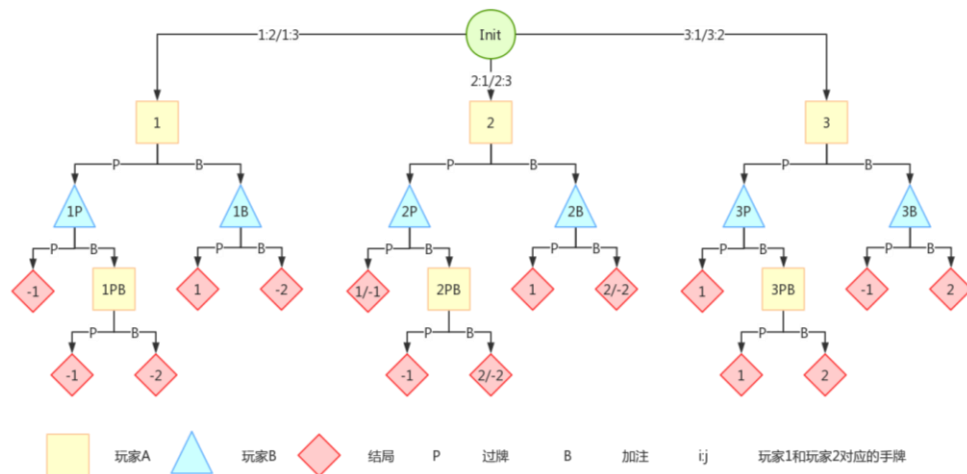
B

加注

$i,j$

玩家1和玩家2对应的手牌

# 库恩扑克 (Kuhn's poker)



- 库恩扑克的信息集有12个： $\{1,1P,1B,1BP,2,2P,2B,2PB,3,3P,3B,3PB\}$
- 每个信息集由不同路径可以到达。如信息集1PB可通过如下路径到达：

1(玩家A拿到大小为1的纸牌)  $\xrightarrow{P}$  1P(玩家A采取过牌行动)  $\xrightarrow{B}$  1PB(玩家B采取加注行动)

可见信息集1PB所对应的行动序列为 $\{P,B\}$

- 在该问题中，到达每个信息集的路径均唯一，因此所有信息集仅对应一个行动序列

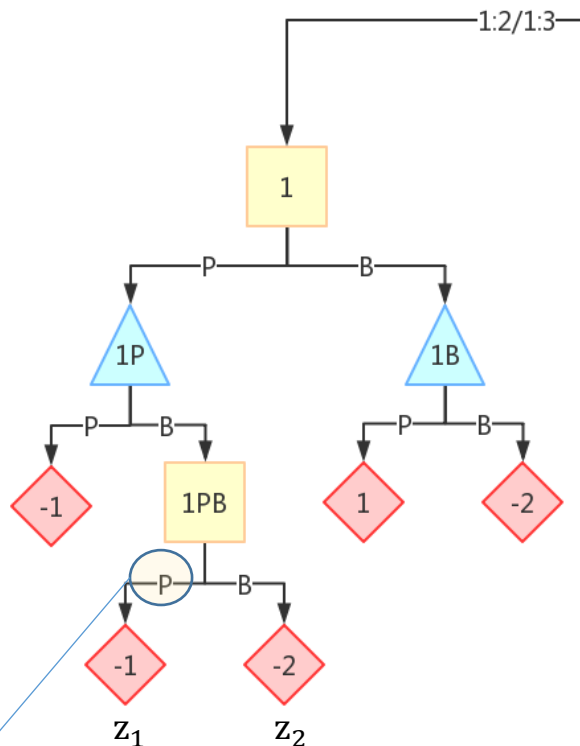


# 库恩扑克（Kunh's pocker）

- 该问题中进行策略选择的算法步骤如下：
  1. 初始化遗憾值和累加策略表为0
  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. 计算每个玩家采取每次行为后的遗憾值
  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. 重复博弈若干次
  7. 根据重复博弈最终的策略，完成最终的动作选择

# 库恩扑克 (Kuhn's poker)

- 假设初始情况下，两个玩家都以随机选择的策略进行决策，即在任一节点，都以50%的概率分别选择过牌和加注
- 若第一轮中，玩家A的博弈过程为 $1 \xrightarrow{P} 1P \xrightarrow{B} 1PB \rightarrow Z_2$ ，收益为 $u_A(Z_2) = -2$
- 计算玩家A针对信息集 $\{1PB\}$ 选择“过牌”行动的遗憾值：
  - 在当前策略下，行动序列 $h = \{P, B\}$ 产生的概率 $\pi_B^\sigma(h) = 1 \times 0.5 = 0.5$
  - 由于在 $\{1PB\}$ 节点选择加注和过牌的概率均为50%，所以当前策略下从行动序列 $h$ 到达终结状态 $z_1$ 和 $z_2$ 的概率分别为： $\pi^\sigma(h, z_1) = 0.5, \pi^\sigma(h, z_2) = 0.5$
  - 又已知 $u_A(z_1) = -1, u_A(z_2) = -2$ ，可知当前策略的虚拟价值：
$$v_A(\sigma, h) = \pi_B^\sigma(h) \times \pi^\sigma(h, z_1) \times u_A(z_1) + \pi_B^\sigma(h) \times \pi^\sigma(h, z_2) \times u_A(z_2)$$
$$= 0.5 \times 0.5 \times (-1) + 0.5 \times 0.5 \times (-2) = -0.75$$
  - 若使用过牌策略，即 $\sigma_{\{1PB\} \rightarrow P}$ ，此时玩家B促使行动序列 $h = \{P, B\}$ 达成的概率仍然为 $\pi_B^\sigma(h) = 0.5$ ，由于最终抵达的终结状态只有 $z_1$ ，所以 $\pi^\sigma(h, z_1) = 1$
  - 则最终选择过牌的虚拟价值为：
$$v_A(\sigma_{\{1PB\} \rightarrow P}, h) = \pi_B^\sigma(h) \times \pi^\sigma(h, z_1) \times u_A(z_1) = 0.5 \times 1 \times (-1) = -0.5$$
  - 在信息集 $\{1PB\}$ 上采取“过牌”的遗憾值
$$r(I, P) = r(h, P) = v_A(\sigma_{\{1PB\} \rightarrow P}, h) - v_A(\sigma, h) = (-0.5) - (-0.75) = 0.25$$



## 库恩扑克（Kunh's pocker）

- 库恩扑克的博弈共有12个信息集，对应上图中的正方形和三角形
- 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

- 对玩家A而言，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	$\alpha$	\	\	0	0	\	\	$1/3 + \alpha$	$3\alpha$	\	\	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。

# 博弈规则的设计

- 问题描述
  - 假设博弈的参与者都是足够理性的
  - 如何设计一个博弈规则能确保公正性或者达到设计者的最大利益
- 挑战
  - 规则复杂
  - 计算量大
- 应用领域
  - 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

## G-S算法 (Gale-Shapley)

- 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
- 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
- 1962年，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，并将其应用于稳定婚姻问题的求解
- **稳定婚姻问题** (stable marriage problem) 是指在给定成员偏好的条件下，为两组成员寻找稳定匹配。由于这种匹配并不是简单地价高者得，所以匹配解法应考虑双方意愿
- 稳定婚姻问题的稳定解是指不存在未达成匹配的两个人都更倾向于选择对方胜过自己当前的匹配对象

## G-S算法：稳定婚姻问题

- 假设有相同数量的单身男性和单身女性，其构成男性集合  $M = \{m_1, m_2, \dots, m_n\}$  和女性集合  $W = \{w_1, w_2, \dots, w_n\}$ 
  1. 单身男性向最喜欢的女性表白
  2. 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
  3. 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
  4. 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。

## 最大交易圈算法（Top-Trading Cycle algorithm）

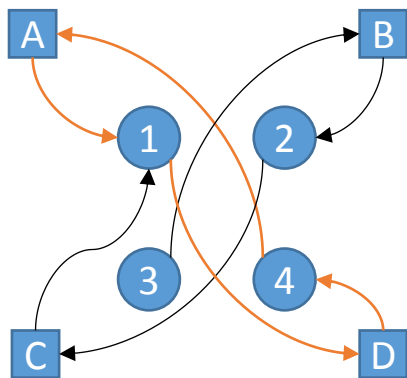
- 在匹配问题中，还有一类交换不可分的标的物的匹配问题，被称为**单边匹配问题**，如远古时期以物易物、或者宿舍的床位分配
- 1974年，沙普利和斯卡夫提出了针对单边匹配问题的稳定匹配算法：最大交易圈算法（TTC），算法过程如下：
  - 首先每个交易者连接一条指向他最喜欢的标的物的边，并从每一个标的物连接到其占有者或是具有高优先权的交易者。
  - 此时形成一张有向图，且必存在交易圈，对于交易圈中的交易者，将每人指向节点所代表的标的物赋予其，同时交易者放弃原先占有的标的物，占有者和匹配成功的标的物离开匹配市场。
  - 接着从剩余的交易者和标的物之间重复进行交易圈匹配，知道无法形成交易圈，算法停止。

## 最大交易圈算法：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4、3、2、1四个床位
- 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

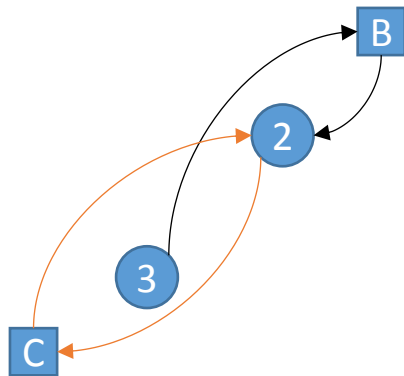


## 最大交易圈算法：室友匹配问题

- 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4、3、2、1四个床位
- 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。

# 提纲

- 1、博弈相关概念
  - 2、遗憾最小化算法
  - 3、虚拟遗憾最小化算法
  - 4、人工智能安全
- 

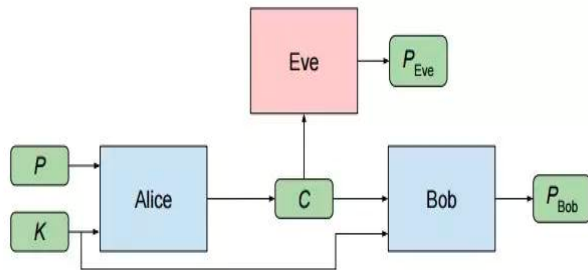
# 基于人工智能的信息安全技术：加密协议

- 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 = 共享密钥，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

# 基于人工智能的信息安全技术：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。

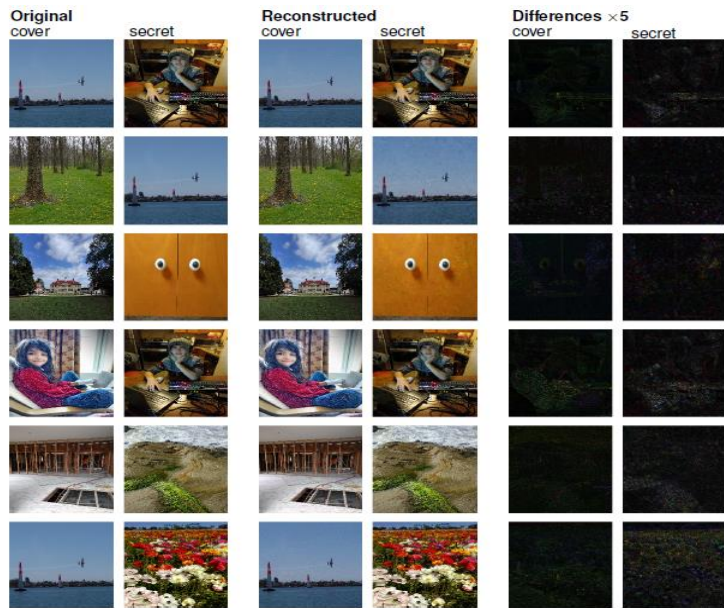


Figure 5: 6 Hiding Results. Left pair of each set: original cover and secret image. Center pair: cover image embedded with the secret image, and the secret image after extraction from the container. Right pair: Residual errors for cover and hidden – enhanced 5 $\times$ . The errors per pixel, per channel 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

# 人工智能的安全：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

# 人工智能的安全：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

## 对抗样本的生成：白盒攻击

- 对人工智能模型的白盒攻击通常会对模型的每一部分进行逐层分解，然后对每一部分添加一定的扰动，使得模型的结果逐步向误判目标类别偏移
- 这是一种非常隐蔽的攻击手段，通过限制扰动的大小可以使得对抗样本看起来与原样本差别很小

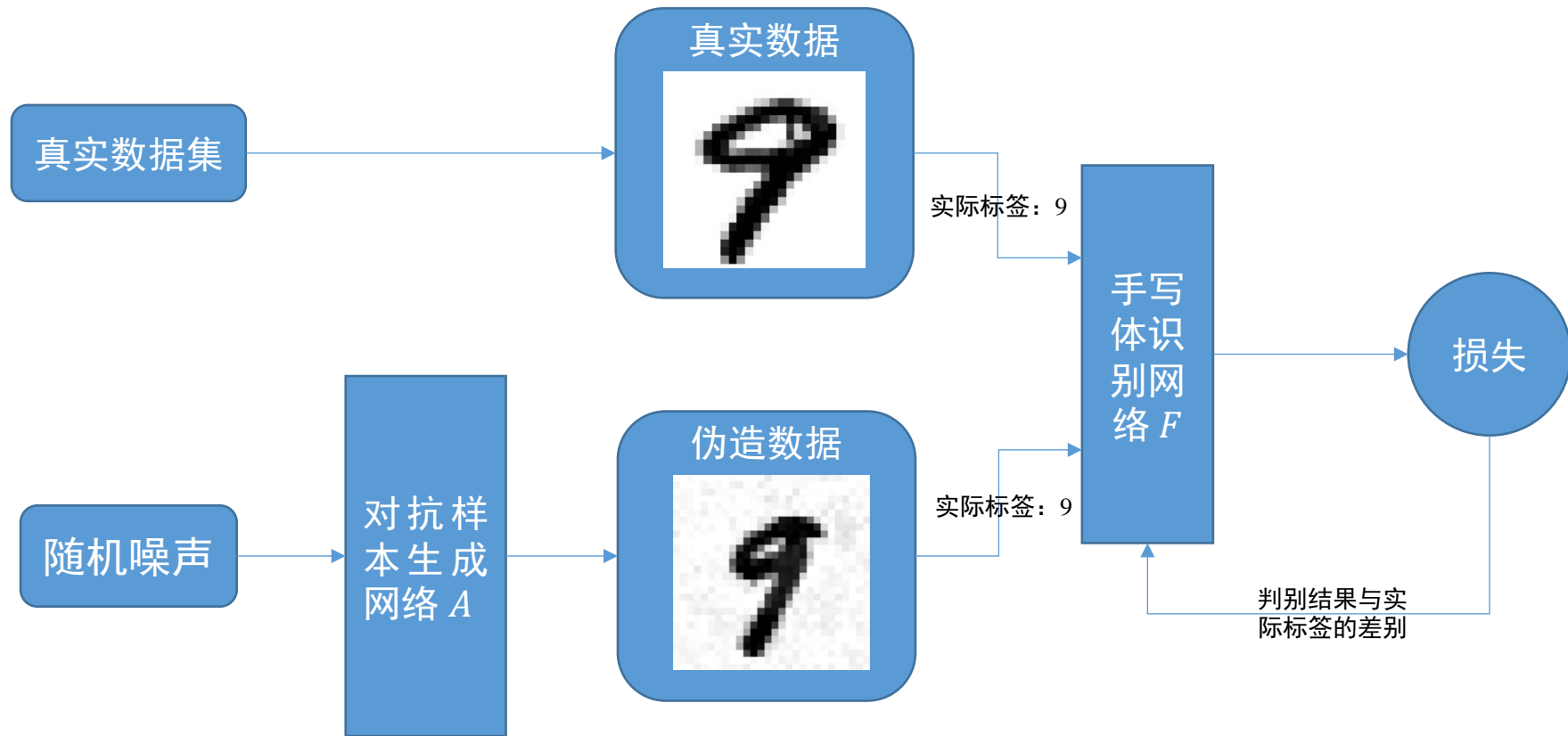


# 白盒攻击的防御策略：生成对抗网络

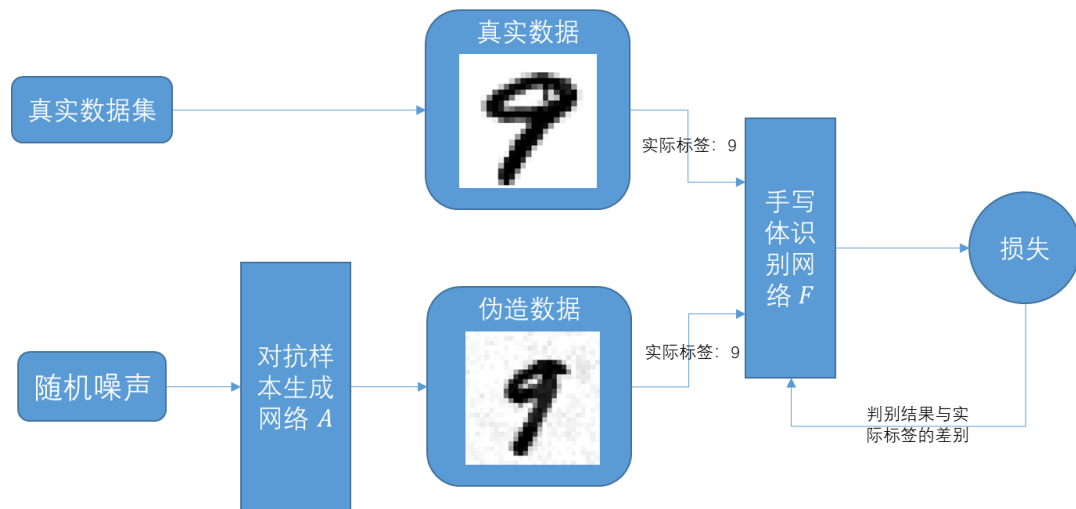
- 由于白盒攻击的过程是对模型内部增添扰动实现的，所以在训练时可以使用同样的方法增强模型训练的鲁棒性
- 如生成对抗网络（generative adversarial network, GAN）就是一种有效的抵御白盒攻击的手段
- 生成对抗网络实际上由两个不同的网络组成：
  - 生成网络：通过神经网络将输入的一个服从简单分布的随机变量转化为能够欺骗判别网络的对抗样本
  - 判别网络：通过神经网络判断输入样本的真实类别
  - 训练时两个网络交替进行参数优化，在对抗过程中共同提升性能
- 在模型训练时，生成网络负责生成对抗样本、判别网络（即我们真正需要的网络）对样本类别进行判断。在这一过程中，生成网络所生成的试图欺骗判别网络的对抗样本会被判别网络识破，从而达到防御白盒攻击的目的



## 白盒攻击的防御策略：判别网络训练过程

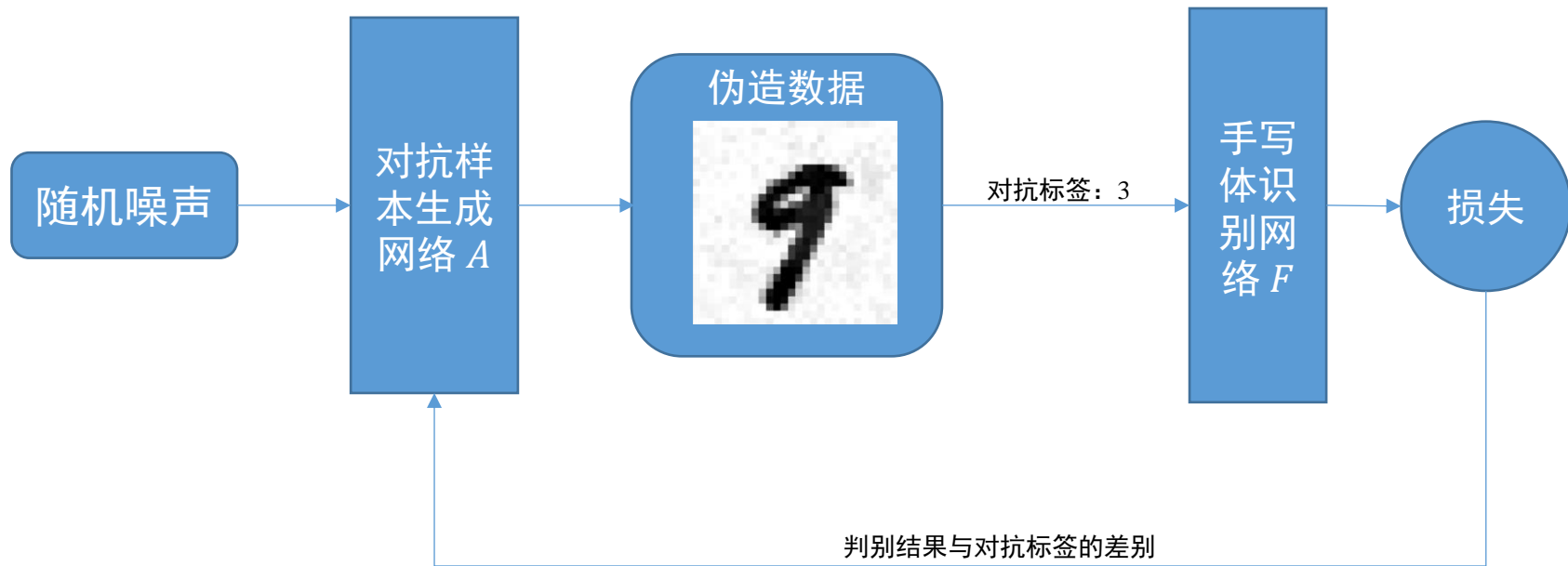


# 白盒攻击的防御策略：判别网络训练过程

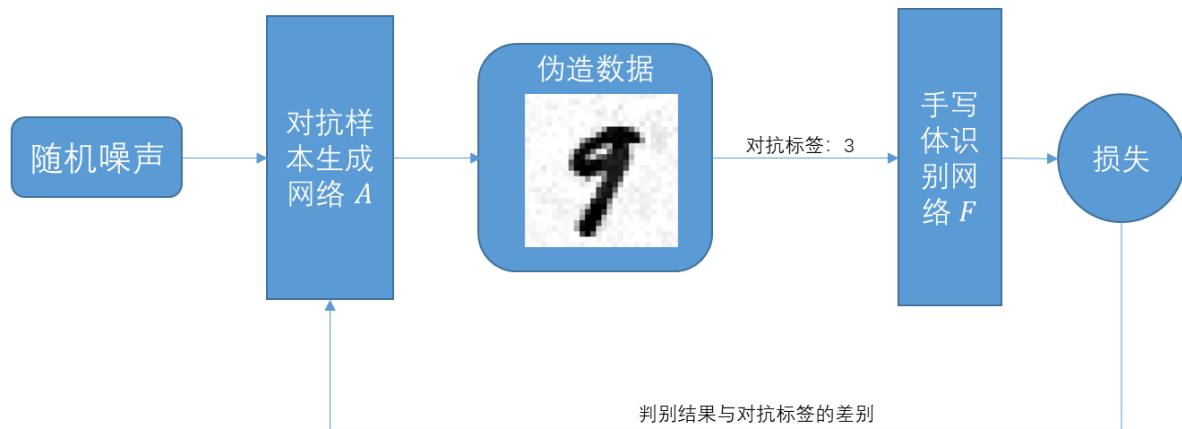


- 判别网络的训练过程分为两个方面
  - 根据真实数据集的数据和其真实标签来增强判别网络识别数据的能力
  - 根据对抗样本生成网络合成的伪造数据来增强判别网络抵抗干扰的能力
- 此时不论是真实数据还是对抗样本，算法都希望判别网络输出结果与图片标签一致。
- 在上述过程中，对抗样本生成网络参数保持不变

## 白盒攻击的防御策略：对抗样本生成网络训练过程



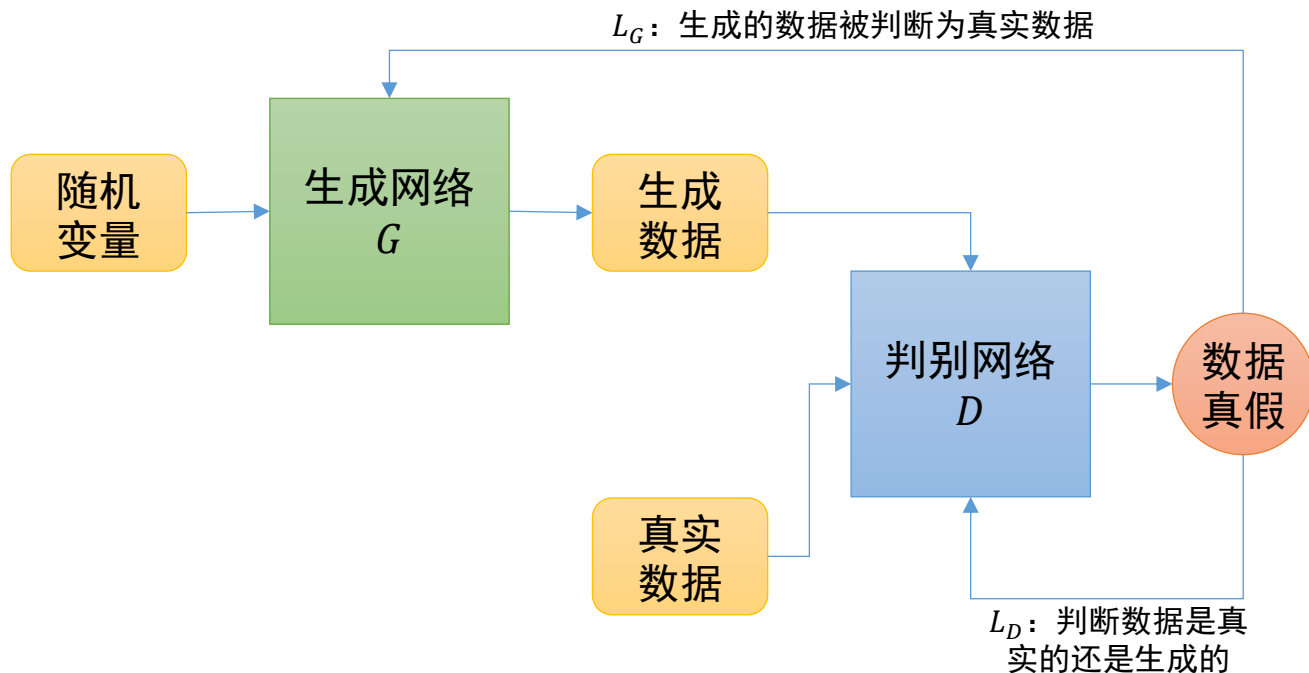
## 白盒攻击的防御策略：生成网络训练过程



- 生成网络的训练过程是判别网络的对抗过程
  - 根据判别网络识别的结果，不断提升对抗样本生成网络合成对抗样本的能力，从而使其能够产生更具有误导性的对抗样本
- 此时希望合成的伪造数据被识别为伪造的对抗标签而不是合成数据所对应的实际标签
- 在该过程中，判别网络参数保持不变。

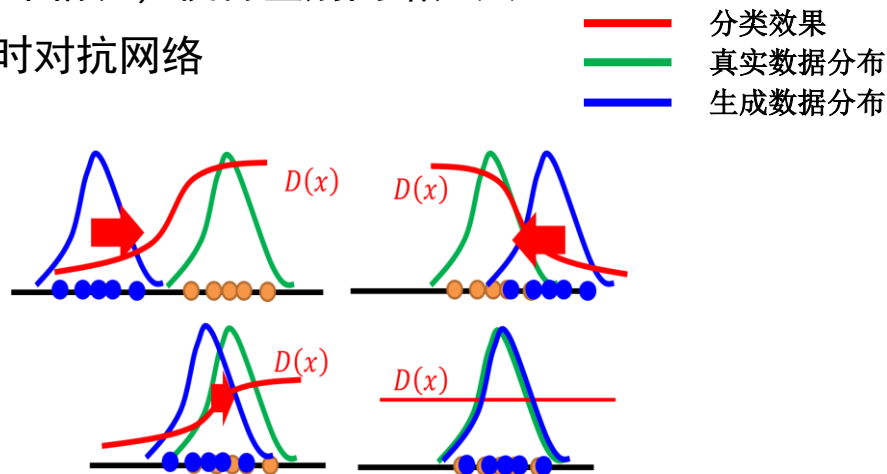
# 生成对抗网络

- 生成对抗网络是深度学习中常用的一种生成模型（generative model）
- 生成对抗网络一般可如下表示：



# 生成对抗网络

- 生成网络通过简单的分布来拟合复杂的分布，并从所拟合的分布中采样得到符合一定要求的样本
- 在训练过程中，判别网络需分辨出真实数据与生成数据的不同、生成网络会逐渐学习得到数据的真实分布情况
- 随着不断优化生成网络，判别网络逐渐无法分辨生成网络所合成数据的真伪
- 最后，生成网络完全模拟出了真实数据的分布情况，使得区别网络无法分辨数据的真伪，开始随机猜测结果，此时对抗网络的训练达到收敛




# 生成对抗网络

生成对抗网络有两个优化目标：生成网络和判别网络。首先优化判别网络模型参数

- $G^* = \operatorname{argmin}_G \max_D V(G, D)$
- $V = \underbrace{E_{x \sim P_{data}} [\log D(x)]}_{\text{判别网络目标}} + E_{x \sim P_G} [\log(1 - D(x = G(z)))]$



- **判别网络目标最大化  $\log D(x)$**
- $P_{data}(x) \log D(x) + P_G(x) \log(1 - D(x))$
- 对  $D(x)$  求导可得最优分类器：
- $D^*(x) = \frac{P_{data}(x)}{P_{data}(x) + P_G(x)}$   最优的分类器能准确区分真假样本

# 生成对抗网络

生成对抗网络有两个优化目标：生成网络和判别网络。接着优化生成网络参数

- $G^* = \underset{G}{\operatorname{argmin}} \max_D V(G, D)$

- $V = E_{x \sim P_{data}} [\log D(x)] + E_{x \sim P_G} [\log(1 - D(x = G(z)))]$



- 生成网络的优化目标是最小化 $\log(1 - D(G(z)))$ ，将最优的判别网络代入这一优化目标

- $$\begin{aligned} V(G, D^*) &= E_{x \sim P_{data}} \left[ \log \frac{P_{data}(x)}{P_{data}(x) + P_G(x)} \right] + E_{x \sim P_G} \left[ \log \frac{P_G(x)}{P_{data}(x) + P_G(x)} \right] \\ &= -2\log 2 + \text{KL} \left( P_{data}(x) \parallel \frac{P_{data}(x) + P_G(x)}{2} \right) + \text{KL} \left( P_G(x) \parallel \frac{P_{data}(x) + P_G(x)}{2} \right) \\ &= -2\log 2 + 2JS(P_{data}(x) \parallel P_G(x)) \end{aligned}$$

- $G^* = \underset{G}{\operatorname{argmin}} D_f(P_{data} \parallel P_G)$

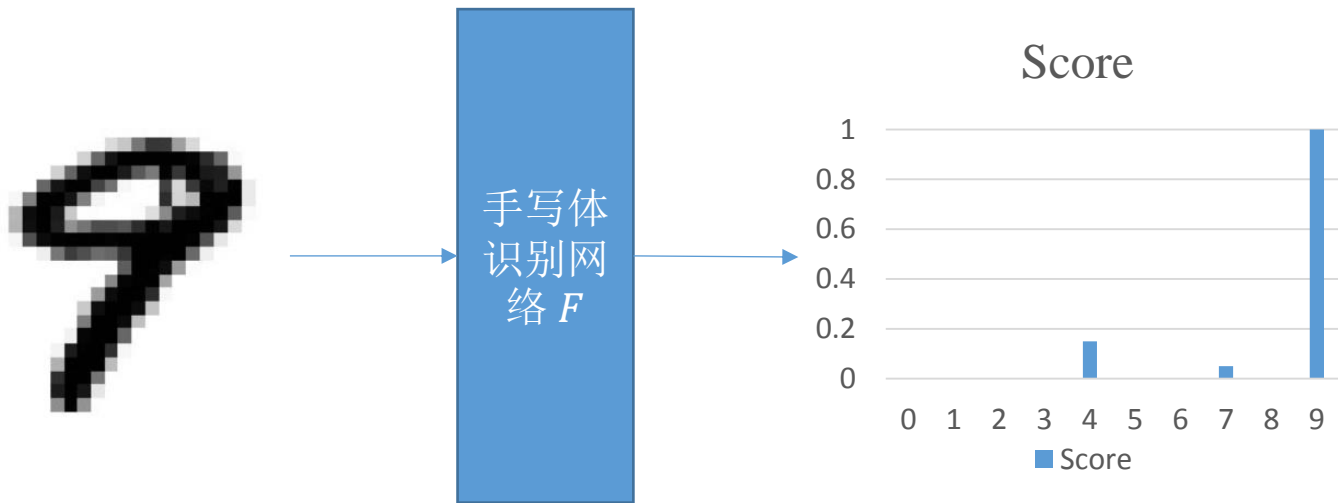


最优生成网络能生成与真实样本具有相同分布的数据



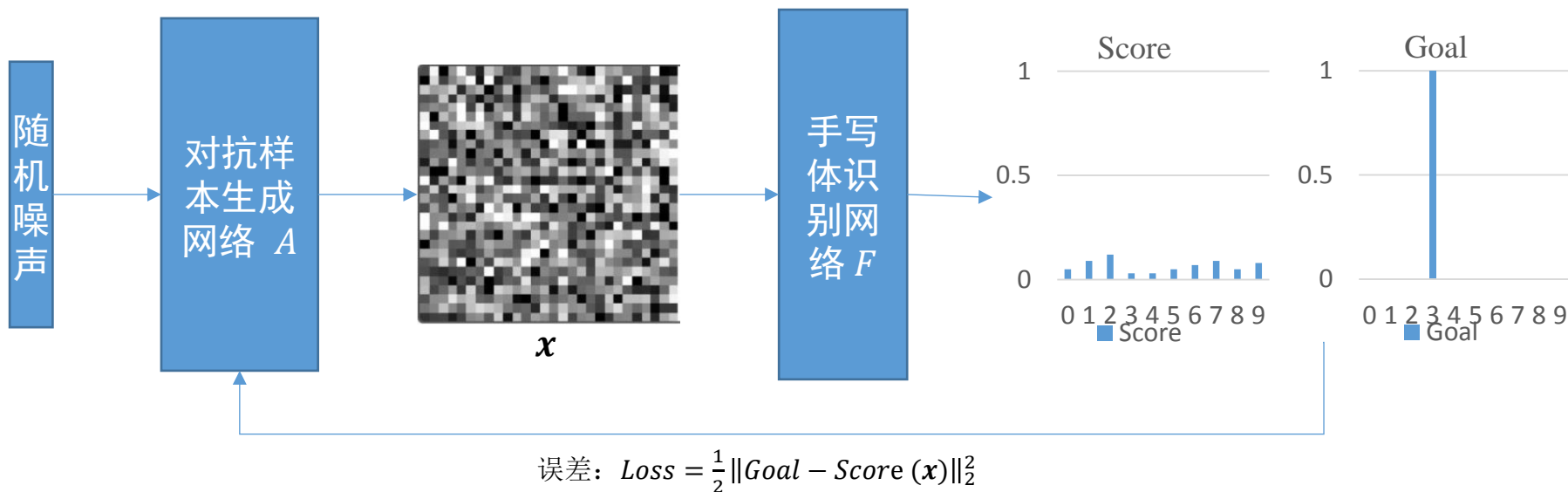
## 对抗样本的生成：无针对攻击（Non-Targeted Attack）

- 无针对攻击：任意生成输入数据，使得模型输出为指定结果
- 假设已经获得一个训练好的神经网络 $F$ ，能够识别手写数字。现在希望生成能够干扰神经网络 $F$ 的对抗样本 $i$ ，使得对抗样本 $i$ 被错误识别为数字3



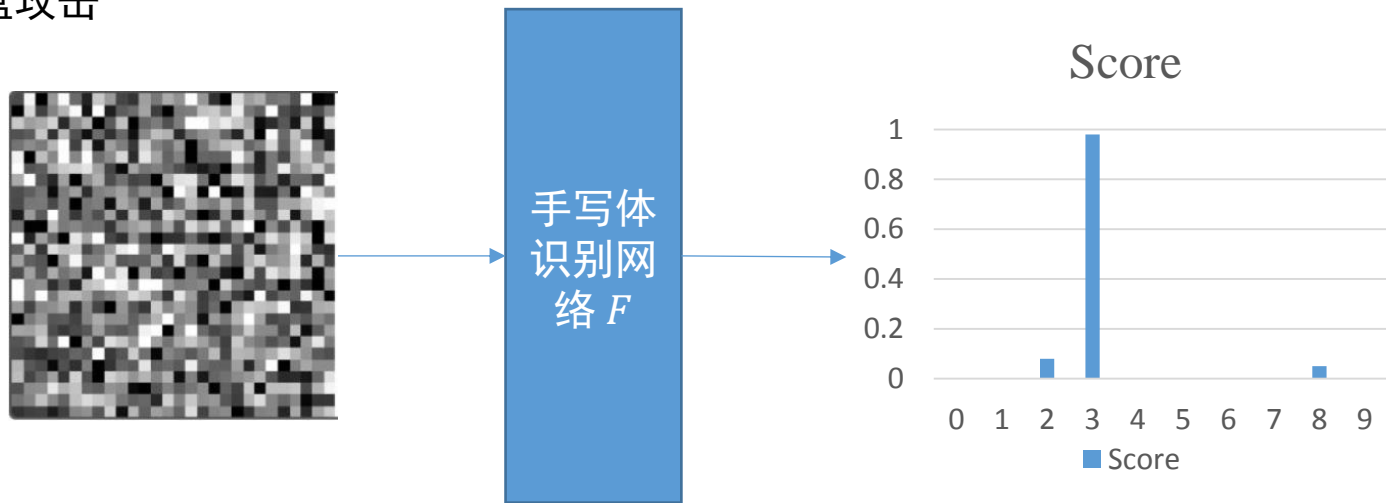
# 对抗样本的生成：无针对攻击（Non-Targeted Attack）

- 训练一个能够生成对抗样本的生成网络 $A$ ，其能够将随机噪声转化为一副对抗样本图片
- 将对抗样本输入手写体识别网络 $F$ ，使用 $F$ 的输出与预设目标之间的误差来优化对抗样本生成网络 $A$



# 对抗样本的生成：无针对攻击（Non-Targeted Attack）

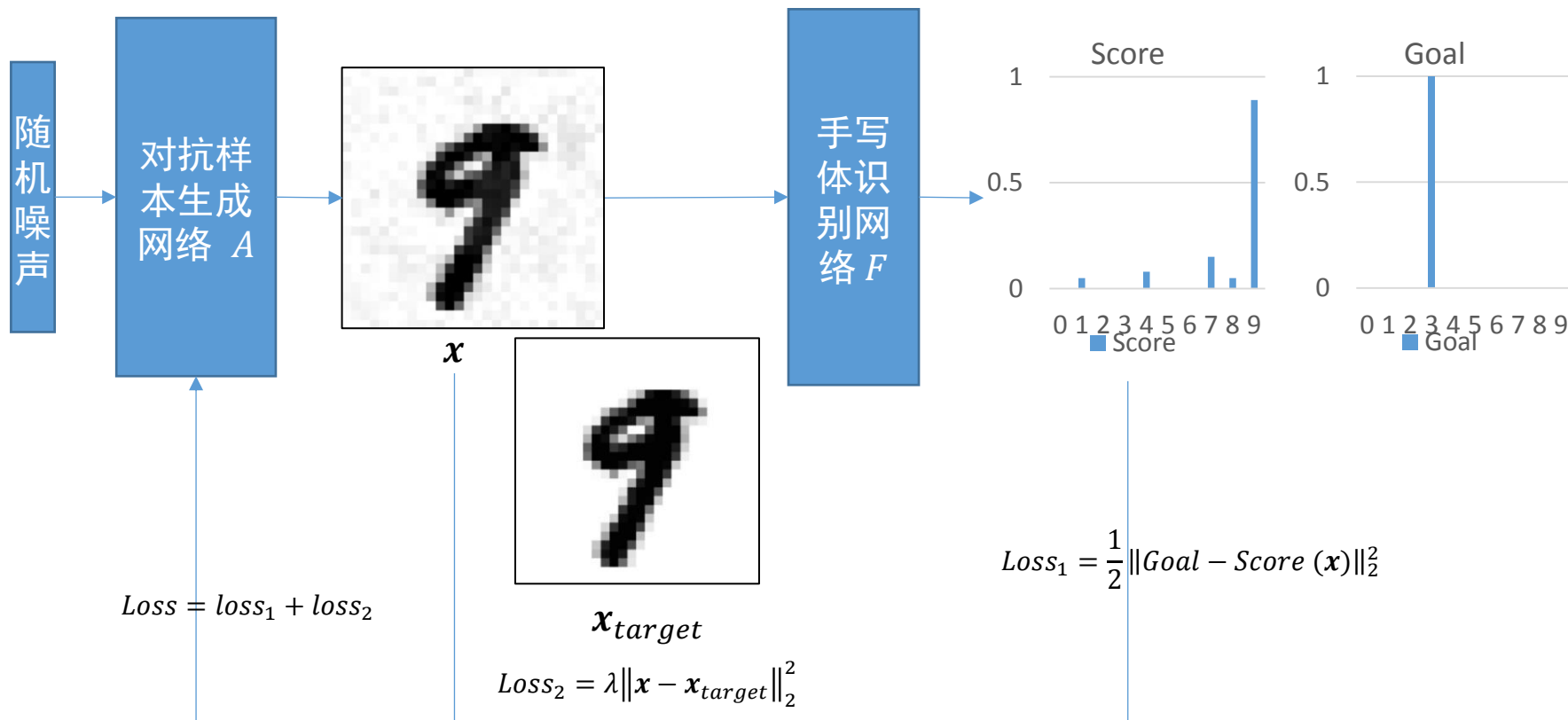
- 通过迭代训练，使得对抗样本生成网络 $A$ 可生成众多被手写体识别网络 $F$ 错误分类为3的对抗样本。
- 这样，手写体识别网络 $F$ 被攻击成功。
- 在对抗样本的生成过程中，没有用到攻击模型的内部结构知识，所以这是一次黑盒攻击



## 对抗样本的生成：有针对攻击（Targeted Attack）

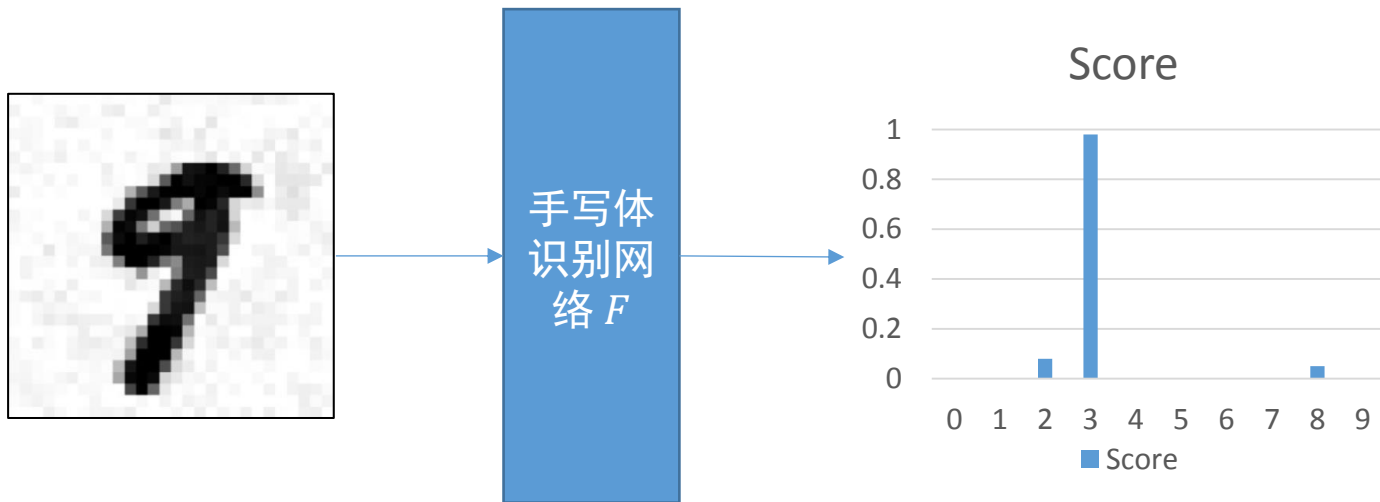
- 有针对攻击：生成人类与模型判断相互迥异的对抗样本
- 假设已经获得一个训练好的神经网络 $F$ ，其能够识别手写体数字，现在想生成能够干扰神经网络 $F$ 的有针对的对抗样本 $i$ 。
- 如：对抗样本 $i$ 被人识别为9，但被 $F$ 错误识别为3。

## 对抗样本的生成：有针对攻击 (Targeted Attack)



## 对抗样本的生成：有针对攻击（Targeted Attack）

- 这种攻击方式同样也是黑盒攻击。可见，手写体识别网络 $F$ 被攻击成功。



# 黑盒攻击的防御策略

- 常用的黑盒攻击防御策略有：
  - 数据压缩：通过对输入数据进行压缩或者降维，在保证识别准确率的情况下提升模型对干扰攻击的鲁棒性
  - 数据随机化：对训练数据进行随机缩放、增强等操作，提升模型的鲁棒性
  - 训练额外的网络来判断训练数据是否为攻击样本