强化学习：

（1）policy-based :基于policy函数的，也就是说每次观测到状态s，根据policy函数计算出来在该状态下不同action的概率，然后通过随机抽样获取一个动作给agent去执行

（2）optimal action-value function based:也就是说每次观测到一个状态s 就根据这个状态s 让Q*函数预测出来不同动作的评分，然后选择评分最高的那个作为action

公式：Q_pai(s_t,a_t) = E[U_t|S_t=s_t,A_t=a_t]

也就是说Q{pai}*是未来reward的一个加权和（U_t）的期望值，*Q{pai}越大，就意味着未来reward的加权和期望越高，说明某种状态s下的某个动作a更加能帮助我们达到目的，动作a更应该被执行

- Action-value function: $Q_\pi(s_t, a_t) = \mathbb{E}\,[U_t | S_t = s_t, A_t = a_t]$.
- For policy $\pi$, $Q_\pi(s, a)$ evaluates how good it is for an agent to pick action $a$ while being in state $s$.
- State-value function: $V_\pi(s) = \mathbb{E}_A\,[Q_\pi(s, A)]$

- $Q^\star(s_t, a_t) = \max_\pi Q_\pi(s_t, a_t)$.

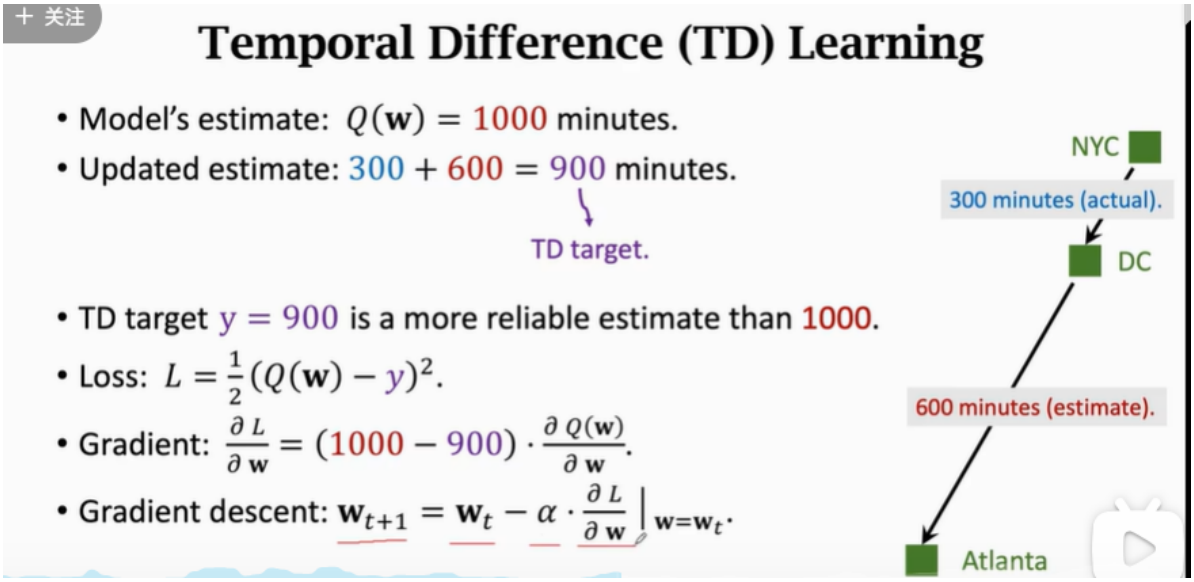
注意状态转移函数是随机的，我们是不知道的，环境会做出选择

Q*函数与policy函数无关，目标是回报期望最大化

Deep Q-Network(DQN):用神经网络近似Q*函数

图像=》卷积层=》特征向量=》全连接层=》对于动作的打分构成的向量（维数是动作空间的大小）

奖励值reward就是**强化学习中的监督信号**



**Identity:** $U_t = R_t + \gamma \cdot U_{t+1}$.

- $U_t = R_t + \gamma \cdot R_{t+1} + \gamma^2 \cdot R_{t+2} + \gamma^3 \cdot R_{t+3} + \gamma^4 \cdot R_{t+4} + \cdots$
  $= R_t + \gamma \cdot (R_{t+1} + \gamma \cdot R_{t+2} + \gamma^2 \cdot R_{t+3} + \gamma^3 \cdot R_{t+4} + \cdots)$
  $= U_{t+1}$

**Identity:** $U_t = R_t + \gamma \cdot U_{t+1}$.

**TD learning for DQN:**

- DQN's output, $Q(s_t, a_t; \mathbf{w})$, is estimate of $\mathbb{E}[U_t]$.
- DQN's output, $Q(s_{t+1}, a_{t+1}; \mathbf{w})$, is estimate of $\mathbb{E}[U_{t+1}]$.
- Thus, $Q(s_t, a_t; \mathbf{w}) \approx \mathbb{E}[R_t + \gamma \cdot Q(S_{t+1}, A_{t+1}; \mathbf{w})]$.
  $Q(s_t, a_t; \mathbf{w}) \approx \mathbb{E}[U_t]$
  $Q(S_{t+1}, A_{t+1}; \mathbf{w}) \approx \mathbb{E}[U_{t+1}]$

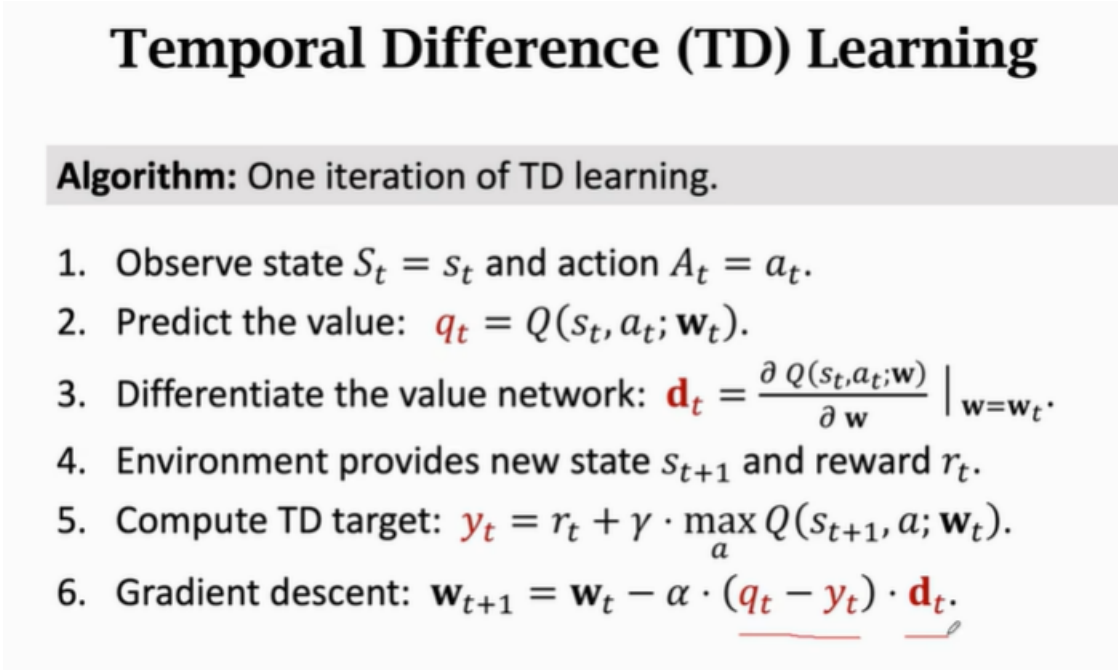
# Train DQN using TD learning

- Prediction: $Q(s_t, a_t; \mathbf{w}_t)$.
- TD target:

$$y_t = r_t + \gamma \cdot Q(s_{t+1}, a_{t+1}; \mathbf{w}_t)$$
$$= r_t + \gamma \cdot \max_a Q(s_{t+1}, a; \mathbf{w}_t).$$

- Loss: $L_t = \frac{1}{2}[Q(s_t, a_t; \mathbf{w}) - y_t]^2$.
- Gradient descent: $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \cdot \frac{\partial L_t}{\partial \mathbf{w}} \Big|_{\mathbf{w}=\mathbf{w}_t}$.

TD算法一轮迭代过程（只要观测到一个r_t就可以进行一轮DQN参数更新）：




==========

**[论文]Deep Reinforcement Learning from Human Preferences.**

**要解决的问题?**

怎样在缺乏reward函数或者这个函数不方便设计的时候去使用强化学习方法?

答案是 让模型产生更符合人期望的轨迹（也就是（state,action）序列）同时做尽可能少的询问

**基本思想:**

根据人类偏好去拟合一个reward 函数并且不断迭代更新这个函数

**数据存储形式:**

以三元组形势来存储（片段1，片段2，偏好分布），其中偏好分布有三种可选{1，0}/ {0，1} /{0.5,0.5}（根据人类给出的偏好标签来做出选择）

**网络更新过程:**

（1）政策函数和环境相互作用 =》 （2）产生轨迹 =》（3）选择轨迹中的两段（每一个都是一个（state,action）序列）并且根据现有的reward_hat去做出偏好预测 =》 （4）让人类去比较选择偏好的一段 =》（5）根据比较结果以及第3步的偏好预测 通过监督学习去优化reward函数

**具体说明:**

**(1)**第三步的偏好预测怎么计算?

$$\hat{P}\left[\sigma^1 \succ \sigma^2\right] = \frac{\exp \sum \hat{r}\left(o_t^1, a_t^1\right)}{\exp \sum \hat{r}\left(o_t^1, a_t^1\right) + \exp \sum \hat{r}\left(o_t^2, a_t^2\right)}.$$

其中的求和是在segment序列的轨迹片段长度上去做的，这个序列由很多不同时间点的（o,t）组成，这里的（o,t）就是我们上面说到的（state,action），即某种状态以及agent在这种状态下选择做出的action（这里是把我们预测的reward_hat函数和传统的RL算法结合然后预测了一个policy函数）

注意这里的reward_hat并不是单个预测器产生的，而是一组预测器通过归一化与求平均产生的

单个预测器的训练过程中，使用了l2正则化和随机失活

**(2)**第五步的优化目标?

$$\operatorname{loss}(\hat{r}) = -\sum_{(\sigma^1,\sigma^2,\mu)\in\mathcal{D}} \mu(1)\log\hat{P}\left[\sigma^1 \succ \sigma^2\right] + \mu(2)\log\hat{P}\left[\sigma^2 \succ \sigma^1\right].$$

最小化上面的交叉熵损失，这里的miu(1)，miu(2)其实是上面数据存储形式部分提过的偏好值在片段1与片段2上的分布

**(3)**询问是怎么被选择的?

会选择各个预测器做出的预测方差最大的片段（也就是预测器最不确定的）去做询问

**(4)**请问怎么根据reward_hat函数和传统的RL算法结合然后预测了一个policy函数?

这里要注意选择的算法对reward_hat函数的改变具有鲁棒性，所以选择了policy gradient methods方法，论文中采用了两种策略梯度方法：A2C和TRPO

**(5)**实验设置：

（1）设置了一组基于人类反馈的有特定数量query的组，设定了三组不同数量query的synthetic oracle（给出的偏好是基于真实reward函数的）以及一组传统的RL算法组

（2）设置了不同实验方法的对照组，包括离线提问（只在训练一开始去收集query）,只采用单个预测期，随机提问，不采用正则化，改变拟合训练目标（将比较替换成真实的reward分数）

实验结果：

有的时候表现的甚至更好因为这种基于人类偏好的好像能更好的去对reward函数进行建模，agent能够通过人类偏好反馈学习到一些创新动作

实验发现有些时候用真实的reward值去替换比较来训练模型效果会不好，原因是具体reward值会急剧变化增加了回归的难度，但是比较法得到的曲线结果较为平滑（可能抗干扰性更好）

传统的训练都是一种自回归的方式（就是预测下一个词的方式），而不是以遵从人类的指令为训练目标的，所以说需要微调

目标有两方面的：（1)很好的遵从人类指令 （2）不产生有害信息，做到有帮助，没有偏见，没有其他恐怖言论等等

step1:使用监督学习方法微调GPT3

监督的微调（自回归方式）

奖励模型大小：6B，更大的话会不稳定

根据这些prompts，生成了三个不同的数据集用于微调过程：

1. SFT（supervised fine-tuning）数据集，使用标记者演示来训练SFT模型，有约13k个训练prompts（来自API和标记者编写）
2. RM数据集，使用模型输出的标记者排名来训练RM，有约33k个训练prompts（来自API和标记者编写）
3. PPO数据集，没有任何人为标记，用作RLHF微调的输入。有约31k个训练prompts（仅来自API）

左边：评估指标是哪个回答更受人类欢迎，人觉得哪个是更好的

被测用户的不同（下面是参与了标注的，上面是未参与标注的人群

横轴，两个不同的API收集的prompt

右边：对于真实性的一个评估（避免瞎编的情况 真实性还包括在一些closed-domain的任务上不胡编乱造（summary），幻觉少）右边：给了一个提示（不知道可以不说）

三条柱子代表不同的模型大小（左右两幅图对比，）

最后：有毒性/安全性的评估（Respectful:给模型一个提示，使它以respectful的语气来回答）

DPO梯度更新的每一步都在增加被人类偏好的数据的生成概率，并降低不被人类偏好的数据的生成概率。同时，这一个更新是加权的，模型的隐式reward model计算的reward误差越大，权重也越大。这个梯度更新在直观上非常容易理解，感觉上就非常正确。文章还测试了如果去掉这个权重，模型性能就会劣化。

(下面的部分还有待补充)

[论文]Aligner: Achieving Efficient Alignment through Weak-to-Strong Correction, 2024

基本思想：

query->unaligned answer -> aligned answer

why not RLHF?

(1)feedbach很难精准的代表人类价值观

(2)需要模型参数可见，不方便应用在调用API的LLM上

aligner:
容易训练 计算资源需求仅依赖于需要达到的aligner的效果，与下层LLM的参数量无关

与残差网络的相似之处：

会保存初始未对齐的回答，

...待补充

基本思想：使用一个能够评估遵守程度的模型去检测其他的ai系统

一些术语：

helpful model:主要目标是提供包含相关信息的答案，而在初始训练中不特别关注避免有害性内容

Red teaming：设计对抗性的提示去测试模型在被诱导的情况下，不产生有害内容的能力（提高模型的鲁棒性和泛化能力）