**Mini-Batch：**

CSE-CIC-IDS2018这种有监督的数据集格式为：[特征1, 特征2, ..., 特征64, 标签]，一组特征是一些ip地址，端口等信息，标签是攻击的类型，但是这种数据格式ddqn无法处理，需要变化[st,at,st+1]这种格式。而特征可以定义为状态，标签可以定义为动作。

随后进行随机取样去训练模型，形成下面图的形式（状态-动作-状态转移过程），这样dqn可以知道自己的前状态动作，通过保留a标签，反馈一个新状态，并通过观察新状态，从而学习a的影响。

图形用户界面, 图示

描述已自动生成

**Q-Learning：**

Q Learning 构建了一个状态-操作值的 Q 表，维度*为 （s， a），*其中 *s* 是状态的数量，*a* 是操作的数量。从根本上说，Q 表将状态和操作对映射到 Q 值。

表格, 日历

描述已自动生成

但是，在实际场景中，状态的数量可能非常多，这使得构建表在计算上变得困难。

**Q-Function：**

图片包含 形状

描述已自动生成

为了解决这一限制，我们使用 Q 函数而不是 Q 表，它实现了将状态和操作对映射到 Q 值的相同结果。

由于神经网络非常擅长对复杂函数进行建模，因此我们可以使用神经网络（我们称之为deep Q Network）来估计这个 Q 函数。

图形用户界面, 应用程序

描述已自动生成

图示

描述已自动生成

1. 这里已知一个状态，通过把它输入到Q网络并得到各动作的Q值，然后用选择一个动作，（贪婪策略会让Q网络保持一定的探索率，后面会逐渐减少探索率），然后环境会输出和，这样就得到了一个experience，然后将这个经验存入经验池中，直到经验池存满。
2. 下面会让Q网络继续学习，但是这个时候Q网络的探索率已经很低了，所以大概率会选择Q值最大的那个动作，按照上述步骤得到experience并替换掉经验池中早期的数据，使其包含较旧和较新样本的混合。
3. 接下来将随机抽取经验池的experience分别输入到两个网络中，Q网络通过和得到预测Q值；Target Q网络则通过得到一个最大的Q值，也就是Target Q值，将Target Q值乘上系数，再加上就是实际Q值。

Target Q值 = γ max(q1,q2,q3,…,qn)

实际 Q值 =  + γ max(q1,q2,q3,…,qn)

1. 使用实际 Q值和预测Q值之间的差值计算均方误差损失

