高级机器学习
作业二

学号, 作者姓名, 邮箱

2018 年 12 月 17 日

1 [30pts] Learning Theory

(1) **[10pts] VC维**

试讨论最近邻分类器假设空间的VC维大小，并给出证明.

(2) **[10pts] Rademaher复杂度**

试证明: 常数函数c的Rademaher复杂度为0.

(3) **[10pts] PAC**

$\mathcal{X} = \mathbb{R}^2, \mathcal{Y} = 0, 1$.假设空间$\mathcal{H}$定义如下：$\mathcal{H} = \{h_r : r \in \mathbb{R}_+\}$,其中$h_r(x) = \mathbb{I}(\| x \| \leq r)$,假定假设空间是可分的，证明$\mathcal{H}$是PAC可学习的，并且样本复杂度为$\frac{log(1/\delta)}{\epsilon}$
(提示：可考虑返回与训练集一致的最小圆的算法)

Proof. 此处用于写证明*(中英文均可)*

2 [30pts] 文档主题模型

在一个新闻数据集上实现文档主题模型(Latent Dirichlet Allocation (LDA)) [1].
我们提供了一个包含8,888条新闻的数据集，请在该数据集上完成LDA算法的使用及实现。

- 数据集下载：新闻数据集.
- 格式：每行是一条新闻.

数据预处理提示：你可能需要完成分词及去掉一些停用词等预处理工作.

(1) **[10pts] 任务#1：使用LDA模型**

A. 选择开源的LDA库（例如：scikit-learn），并在提供的数据集上学习使用.

B. 给出$K = \{5, 10, 20\}$个主题时，每个主题下概率最大的$M = 10$个词及其概率.

(2) **[20pts] 任务#2：实现LDA模型**

A. 不借助开源库，自己完成LDA算法.

B. 给出$K = \{5, 10, 20\}$个主题时，每个主题下概率最大的$M = 10$个词及其概率.

Solution.

3 [40pts] 强化学习实验

用DQN (deep Q Networks) 训练Flappy Bird. 请各位同学根据DQN算法流程，补全提供的代码包中deep_q_networkd.py文件中“# TODO”部分代码(补全epsilon-greedy action selection以及Q learning updating)，了解DQN算法，并进行训练，本实验时间相对较久.

本次实验所需要的依赖如下：

- python2.7 or python3;
- pygame;
- OpenCV-python;
- TensorFlow (建议使用1.1-1.6).

强化学习中经典的off-policy算法Q-Learning的原始版本采用表格形式来记录Q函数，显然只能应用于有限离散状态、有限离散动作且状态、动作数量较少的情况下，即有维度灾难问题(表格大小正比于$|S| * |A|$). 采用函数近似法，假定Q函数可由状态特征经过某个函数的映射到对应动作的评价值上，可扩大Q-Learning使用范围. 近年来，DeepMind结合深度模型强大的表达能力，用深度神经网络作为近似函数来表达强化学习中的Q 函数，进一步扩大了Q-Learning可用范围. DQN中采用experience replay和target network两种技术，使DQN的训练更加高效且鲁棒，并在atari的部分游戏上取得了人类水平的表现.

DQN的流程大致如下 1：

上图是15年DeepMind发表在Nature上文章中所采用的算法流程，包含了experience replay和target network技术，本次实验不要实现target network ，仅需要实现experience replay即可(实现target network可额外获得5pts bonus). 感兴趣的同学可参阅DQN相关教程或文章，进一步了解两种技术.

本次实验中状态太输入为raw pixel，转为$80 * 80$的灰度图(采用openCV转换)，并将历史最近3个frame叠加到当前frame中作为状态输入，即每一步输入状态为$4 * 80 * 80$，动作为2维离散动作(上、下，action为2维one-hot编码). 网络模型已经搭建好(采用TensorFlow 搭建)，输入为$4 * 80 * 80$，输出为2，对应每个动作对应的Q 值。如下图所示 1.

游戏环境中，单步奖励为0.1，越过一个管道+1，死亡得到−1的惩罚. 可采用其他深度学习框架，如pytorch、keras 等搭建模型并完成训练代码. DQN算法设置可采用如下配置：

- GAMMA = 0.99 # decay rate of past observations;
- OBSERVE = 10000. # timesteps to observe before training;
- EXPLORE = 2000000. # frames over which to anneal epsilon;
- FINAL_EPSILON = 0.0001 # final value of epsilon;
- INITIAL_EPSILON = 0.1 0.2 # starting value of epsilon;
- REPLAY_MEMORY = 50000 # number of previous transitions to remember;
- BATCH = 32 # size of minibatch;
- FRAME_PER_ACTION = 1.

Algorithm 1 DQN with experience replay

Initialize replay memory D to capacity N

Initialize action-value function Q with random weights θ

Initialize target action-value function $\hat{Q}$ with weights $\theta^- = \theta$

for *episode* $= 1, M$ **do**

Initialize sequence $s_1 = x_1$ and preprocessed sequence $\phi_1 = \phi(s_1)$

for $t = 1, T$ **do**

With probability ϵ select a random action a_t

otherwise select $a_t = \arg\max_a Q(\phi(s_t), a; \theta)$

Execute action a_t in emulator and observe reward r_t and image x_{t+1}

Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$

Store transition $(\phi_t, a_t, r_t, \phi_t)$ in D

Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D

Set

$$f(x) = \begin{cases} r_j & \text{if episode terminates at step } j+1 \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j=1}, a'; \theta^-) & \text{otherwise} \end{cases} \tag{3.1}$$

Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ with respect to the network parameters θ

Every C steps reset $\hat{Q} = Q$

end for

end for

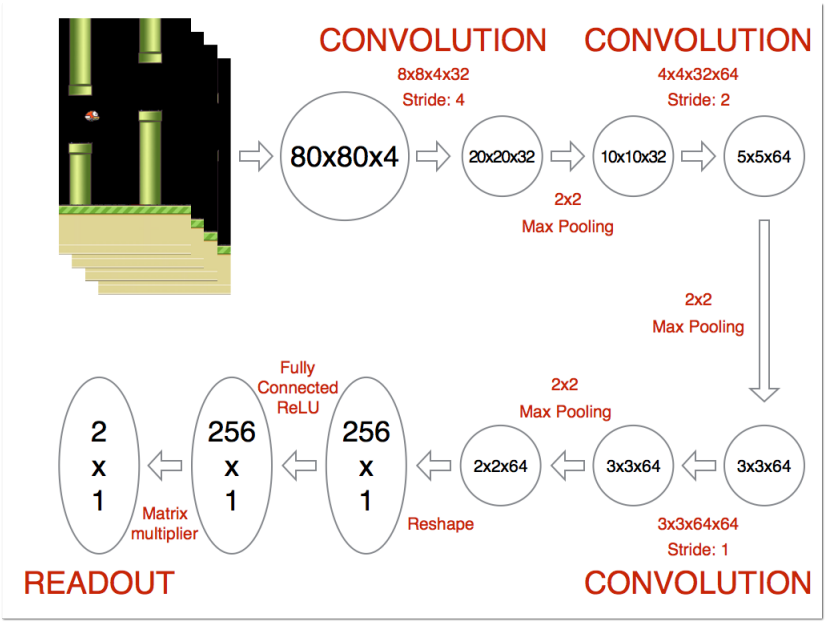

Figure 1: 网络模型.

默认一直训练不会终止，每10,000 frames保存一个模型，默认最大保存5个，保存的模型可恢复用来测试，默认保存在save_model 目录下. 采用GPU可加速训练，仅使用双核CPU训练时，采用如上配置，总样本量到1M (1,000,000个state) 需要时间为20 24h，大概3M可训练出相当不错的策略，考虑到计算咨询和时间，可自行选择训练量.

采用其他深度学习框架时，只需要保持从环境中获得返回的状态、奖励信息，以及是否终止，并可在环境中执行action (再次注意，action为2维one-hot编码). Agent与环境交互过程如下所示：

- sys.path.append("game/");
- import wrapped_flappy_bird as game # import game environment;
- game_state = game.GameState() # initialize;
- # execute an action and get info from the environment;
- x_t, r_0, terminal = game_state.frame_step(action).

本实验提交要求：

仅需提供补全后deep_q_network.py文件，以及训练后的短视频(连续飞行5 – 10s 即可) 或图片或gif动图等辅助证明材料，并说明训练使用样本量. 如果有任何修改或补充说明，请一并说明. (建议写Readme文件或报告)

Solution. *此处用于写解答(中英文均可)*

References

[1] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.