# ECSE526 Assignment 3 Report

David Newton
Kevin Ra

## 1.0 Generalization Between Similar States

### 1.1 Approach

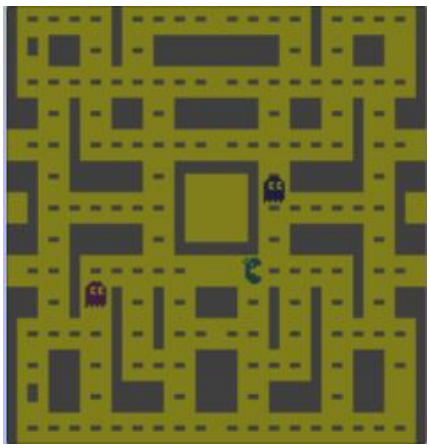
For the project we have chosen to use function approximation with an adaptive exploratory function. We are using the  positions of pacman and each of the ghosts, as well as, the velocity of the ghosts as our features.

#### 1.1.1 State Extraction

We represent the states in terms of the pixel position of Pacman and four ghosts and velocity of the four ghosts. This information can be obtained by getting the screen information through ALE (Arcade Learning Environment) interface.

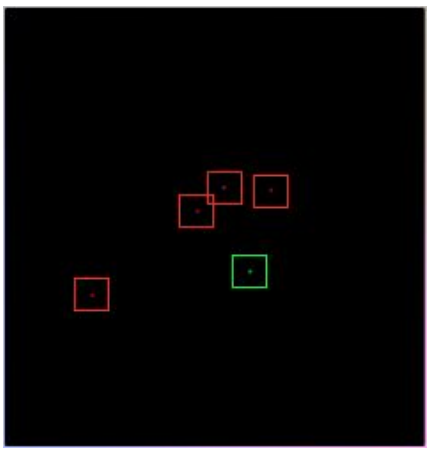
The colors of Pacman and the four ghosts are fixed value so the all pixels that belongs to each object (Pacman or ghosts) can be aggregated. Their center is calculated by taking the average of each coordinate value.

Velocity of the ghosts can be easily determined if we store the previous position and compare it against current position. It's worth mentioning that it's common to not have all ghosts on a given frame. Thus, in order to take the velocity, we need to keep track of how many frames has it passed since specific ghost's last presence on the frame. Figure 1 shows the steps of getting the pixel position of each object.



(a)

(b)



(c)

**Figure 1**. (a) Reference image (b) All pixels except those belong to objects in interest are set to black. Note that there is less than four ghosts in the frame (c) Center of each objects. Pacman's position is marked in green dot and ghosts' position are marked in red dot. Boxes are drawn to help identify the dots. Note that the red dots that doesn't belong to any ghosts in (b) are the center of the ghosts that are not shown in the frame.

1.1.2. Generalization

We are using the function approximation to generalize over the states. The learning rate, **α** is empirically determined by choosing the value that gives the highest score. We found out that having large **α** makes the weights of the function approximation explode. Through experiments we found that low values of alpha worked best. Specifically we settled on a value of alpha = 0.0001 which assured a good balance between underfitting and overfitting.

1.2 Experimental Results

## 2.0 Exploration Function

2.1 Approach

We introduce randomness to prevent the agent from being greedy. If generated random number is less than ρ, we choose random action. Otherwise, we choose an action that gives the best Q value. ρ is inversely proportional to the number of episodes so the agent will become more greedy over time. Initially, ρ is set to 0.1. Results from this adaptive exploration function can be seen below. It seems to reduce the predicted error rate of the utility function faster than the non-adaptive version presented in figure 1 below. This makes sense because as training progresses the function learns more of what the real utility function could be and needs random search less and less.

2.2 Experimental Results

||||||||||||||||Episode 0 ended with score: 190

ERROR = 32.6038 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0765645 + + 0.0327774*f0 + -0.471415*f1 + -0.255671*f2 + 0.110994*f3 + 0.104162*f4 + 0.140347*f5 + -0.207783*f6 + 0.301744*f7 + 0.323375*f8 + -0.172197*f9 + 0.654223*f10 + 0.960313*f11 + 0.857568*f12 + 0.293716*f13 + 0.356878*f14 + 0.460988*f15 + 0.759521*f16 + 0.516479*f17

||||||||||||||||Episode 100 ended with score: 90

ERROR = 1.71333 ALPHA = 1e-06 EXPLORATION RATE = 1.07151e-159

Approximated Utility Function:
U(s) = 0.0754318 + + 0.190012*f0 + -0.324989*f1 + -0.0852298*f2 + 0.0824517*f3 + -0.108221*f4 + 0.0229599*f5 + 0.00281197*f6 + 0.123313*f7 + 0.0363698*f8 + -0.0437527*f9 + 0.666901*f10 + 0.948244*f11 + 0.838243*f12 + 0.26278*f13 + 0.331661*f14 + 0.434726*f15 + 0.717806*f16 + 0.501192*f17

||||||||||||||||Episode 179 ended with score: 90

ERROR = 1.60285 ALPHA = 1e-06 EXPLORATION RATE = 0

Approximated Utility Function:
U(s) = 0.0752013 + + 0.178659*f0 + -0.293645*f1 + -0.0818567*f2 + 0.0674766*f3 + -0.102619*f4 + 0.0177105*f5 + 0.00531666*f6 + 0.144068*f7 + 0.015033*f8 + -0.0458564*f9 + 0.662632*f10 + 0.939042*f11 + 0.818738*f12 + 0.251821*f13 + 0.320287*f14 + 0.414717*f15 + 0.701189*f16 + 0.490312*f17

||||||||||||||||Episode 226 ended with score: 90

ERROR = 1.55936 ALPHA = 1e-06 EXPLORATION RATE = 0

Approximated Utility Function:
U(s) = 0.075094 +  + 0.170912*f0 + -0.277899*f1 + -0.082151*f2 + 0.0588406*f3 + -0.0995153*f4 + 0.0176706*f5 + 0.00652021*f6 + 0.153308*f7 + 0.00825114*f8 + -0.0469002*f9 + 0.659435*f10 + 0.93402*f11 + 0.807515*f12 + 0.245973*f13 + 0.314504*f14 + 0.40271*f15 + 0.691688*f16 + 0.484307*f17

||||||||||||||||Episode 290 ended with score: 90

ERROR = 1.50936 ALPHA = 1e-06 EXPLORATION RATE = 0

Approximated Utility Function:
U(s) = 0.074965 +  + 0.161108*f0 + -0.259268*f1 + -0.0819414*f2 + 0.0481972*f3 + -0.0952243*f4 + 0.0179013*f5 + 0.00803074*f6 + 0.162666*f7 + 0.0011821*f8 + -0.0477316*f9 + 0.654526*f10 + 0.92763*f11 + 0.792915*f12 + 0.238353*f13 + 0.307334*f14 + 0.386497*f15 + 0.678959*f16 + 0.476488*f17

||||||||||||||||Episode 499 ended with score: 90

ERROR = 1.3918 ALPHA = 1e-06 EXPLORATION RATE = 0

Approximated Utility Function:
U(s) = 0.0746216 +  + 0.136147*f0 + -0.21626*f1 + -0.0779159*f2 + 0.0212335*f3 + -0.0818603*f4 + 0.0192681*f5 + 0.0111831*f6 + 0.177893*f7 + -0.0126186*f8 + -0.0472864*f9 + 0.635149*f10 + 0.909267*f11 + 0.749106*f12 + 0.215203*f13 + 0.287476*f14 + 0.335717*f15 + 0.63881*f16 + 0.452997*f17

## 3.0 Agent Performance

### 3.1 Results

The annotations and graph below show that, for an alpha = 0.001 and a random exploration rate of 0.1, improvements in the score and error rate of the utility prediction function was significant in just 1000 training runs.  These 1000 runs took about 5 minutes to run, so it is conceivable that the estimated utility function could be trained with several more thousand runs before time would become an issue.

Annotations:

||||||||||||||||Episode 1 ended with score: 240

ERROR = 9.7459 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0763109 +  + 0.0732936*f0 + -0.4631*f1 + -0.25748*f2 + 0.133734*f3 + 0.0767457*f4 + 0.131839*f5 + -0.19302*f6 + 0.198667*f7 + 0.270716*f8 + -0.123726*f9 + 0.652954*f10 + 0.958914*f11 + 0.856282*f12 + 0.289843*f13 + 0.35693*f14 + 0.459959*f15 + 0.756399*f16 + 0.517074*f17

||||||||||||||||Episode 2 ended with score: 190

||||||||||||||||Episode 87 ended with score: 260

ERROR = 2.42364 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0757692 +  + 0.0116469*f0 + -0.0318165*f1 + 0.0289035*f2 + 0.0048526*f3 + 0.0158699*f4 + 0.00199983*f5 + -0.0220411*f6 + -0.0243991*f7 + -0.00426899*f8 + 0.028375*f9 + 0.618593*f10 + 0.895295*f11 + 0.778679*f12 + 0.245451*f13 + 0.331083*f14 + 0.424897*f15 + 0.689803*f16 + 0.48883*f17

||||||||||||||||Episode 191 ended with score: 200

ERROR = 2.28832 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0749822 +  + 0.0089188*f0 + -0.0345611*f1 + 0.00679856*f2 + 0.00775385*f3 + 0.0137833*f4 + 0.000153896*f5 + -0.0112706*f6 + -0.00109369*f7 + 0.00350004*f8 + 0.00886828*f9 + 0.565185*f10 + 0.824915*f11 + 0.685568*f12 + 0.21153*f13 + 0.285247*f14 + 0.390951*f15 + 0.632093*f16 + 0.462544*f17

||||||||||||||||Episode 300 ended with score: 110

ERROR = 1.87103 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0743297 +  + -0.0496336*f0 + -0.0090359*f1 + 0.016648*f2 + 0.0165145*f3 + 0.0173656*f4 + -0.0125044*f5 + -0.0112963*f6 + 0.0124986*f7 + 0.00538696*f8 + 0.0283922*f9 + 0.501795*f10 + 0.779097*f11 + 0.599058*f12 + 0.181347*f13 + 0.23338*f14 + 0.353012*f15 + 0.558779*f16 + 0.42952*f17

||||||||||||||||Episode 400 ended with score: 200

ERROR = 1.73577 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0750459 +  + -0.0237195*f0 + -0.0193432*f1 + 0.018496*f2 + 0.0378793*f3 + 0.00546328*f4 + 0.0133019*f5 + -0.0254532*f6 + 0.00774356*f7 + -0.00245984*f8 + 0.0150211*f9 + 0.447597*f10 + 0.735102*f11 + 0.533491*f12 + 0.151899*f13 + 0.19216*f14 + 0.318307*f15 + 0.493628*f16 + 0.387891*f17

||||||||||||||||Episode 500 ended with score: 440

ERROR = 2.49566 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0753916 +  + -0.0276712*f0 + -0.00893359*f1 + 0.00818735*f2 + 0.0288096*f3 + -0.00156974*f4 + -0.00825196*f5 + -0.0171549*f6 + 0.00280028*f7 + -0.00745925*f8 + 0.0286639*f9 + 0.399339*f10 + 0.699117*f11 + 0.481707*f12 + 0.126427*f13 + 0.151305*f14 + 0.289371*f15 + 0.438163*f16 + 0.353832*f17

||||||||||||||||Episode 501 ended with score: 180

||||||||||||||||Episode 700 ended with score: 120

ERROR = 1.32846 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0757817 +  + -0.0343083*f0 + -0.00184315*f1 + 0.0171964*f2 + 0.0357199*f3 + -0.00291668*f4 + 0.00591314*f5 + -0.00223229*f6 + -0.00814915*f7 + -0.00483496*f8 + 0.00836334*f9 + 0.31821*f10 + 0.62976*f11 + 0.388928*f12 + 0.0865806*f13 + 0.0978647*f14 + 0.238517*f15 + 0.343079*f16 + 0.298652*f17

||||||||||||||||Episode 800 ended with score: 190

ERROR = 1.436 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0761858 +  + -0.0268762*f0 + -0.00559379*f1 + 0.00891035*f2 + 0.0253722*f3 + 0.0260201*f4 + -0.0170631*f5 + -0.0155441*f6 + 0.00305715*f7 + -0.0244218*f8 + 0.0299521*f9 + 0.27822*f10 + 0.592918*f11 + 0.351371*f12 + 0.0707155*f13 + 0.0741906*f14 + 0.215556*f15 + 0.300931*f16 + 0.264149*f17

||||||||||||||||Episode 999 ended with score: 380

ERROR = 2.37068 ALPHA = 1e-06 EXPLORATION RATE = 0.1

Approximated Utility Function:
U(s) = 0.0768138 +  + -0.0207544*f0 + 0.00228381*f1 + 0.0240703*f2 + 0.0199585*f3 + -0.00381495*f4 + -0.00646814*f5 + -0.00600963*f6 + 0.0022969*f7 + -0.0247613*f8 + 0.0162521*f9 + 0.215904*f10 + 0.526179*f11 + 0.290773*f12 + 0.0434999*f13 + 0.0407054*f14 + 0.175865*f15 + 0.227824*f16 + 0.216542*f17

Below are the graphs that shows how our errors and scores are changing over the number of training episodes. The error is calculated by the average difference between observed utility and predicted utility at $n^{th}$ episode (Figure X). Figure Y shows the total score the agent obtained on $n^{th}$ episode. It can be shown that the errors and decreasing and scores are increasing as the number of episode increases.

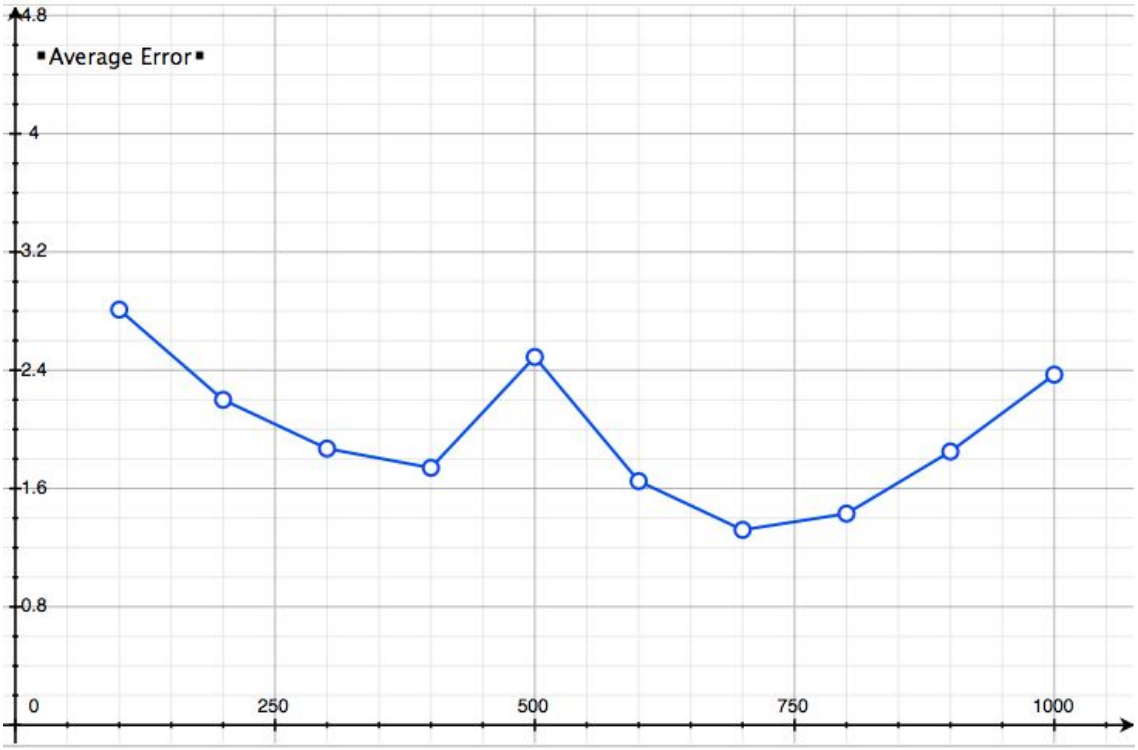



**Figure 1**. Number of episodes vs. average error

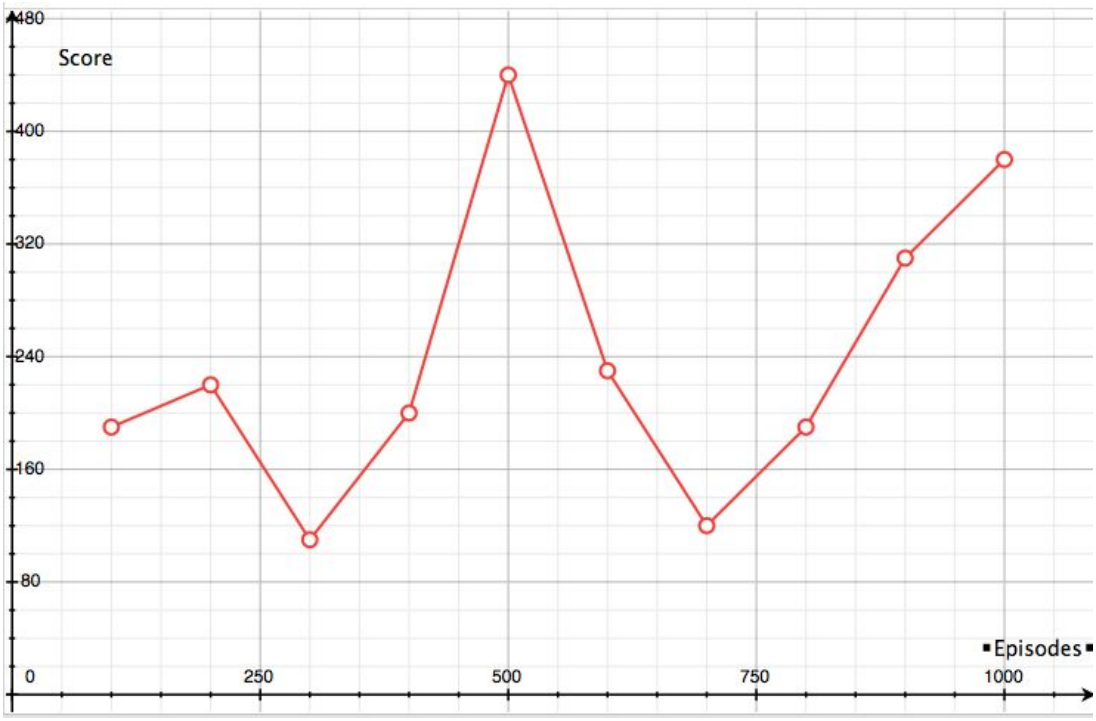



**Figure 2**. Number of episodes vs. score