Programming Assignments: CECS 451

Darin Goldstein

1 Jugs and water

What has 4 legs and is always ready to travel?

The purpose of this assignment is to illustrate the power and limits of AI search. In the movie *Die Hard with a Vengeance*, McClane and Zeus need to disarm a bomb in a park. They are given an unlimited supply of water and two jugs with capacity 3 and 5 gallons respectively. In order to save the day, they need to get *exactly* 4 gallons of water into the 5-gallon jug. (They have an unlimited supply of water coming from a fountain.) McClane initially wants to pour 3 gallons into the 5-gallon jug and estimate the rest, but together they manage to find a way to get 4 gallons into the 5-gallon jug without using any estimating. Try it yourself before reading one possible solution below...

1. Fill the 5-gallon jug.
2. Pour 3 gallons from the 5-gallon jug into the 3-gallon jug.
3. Empty all water in the 3-gallon jug.
4. Pour 2 gallons from the 5-gallon jug into the 3-gallon jug.
5. Fill the 5-gallon jug.
6. Pour 1 gallon from the 5-gallon jug into the 3-gallon jug (filling the 3-gallon jug).
7. Empty the 3-gallon jug. The only water remaining in the jugs is 4 gallons in the 5-gallon jug.

Your goal will be to devise an artificial intelligence technique that determines whether a given water jug problem can be solved and, if so, solves it using the minimum number of possible moves; we don't want the bomb to explode while McClane and Zeus are busy pouring the water.

Assume that the bad guy is in (predictably) a bad mood. Rather than give McClane and Zeus only 2 jugs, he decides to give them 4. (You are guaranteed that the maximum jug size is 45 units.) The goal is to leave a predetermined number of units of water in the largest jug and no other water in any other jug using the minimum number of valid moves (defined below).

A *move* is defined to be exactly one of the following:

1. Fill up a jug to the top using water from the fountain.
2. Pour out all the contents of a jug into the fountain.
3. Assuming you choose a source jug and destination jug, you must pour water from the source to the destination until either (i) the source jug runs out of water or (ii) the destination jug fills up to the top (whichever comes first).

Determine the minimum number of valid moves to get from the configuration where all jugs are empty to the configuration where all jugs are empty except for the largest.

1.1 Input, output

Your program should be given the capacity of each of 4 jugs and the goal in addition to the name of the output file. After computations are complete, you need to write the instructions for how to get the goal amount into the largest jug to the file using the following format. Number the jugs 0, 1, 2, and 3 from smallest capacity to largest. The fountain will be labelled -1. To indicate that jug i should be filled, the instruction is $(-1, i)$ indicating that water is to poured from the fountain into jug i. To indicate that jug i should be emptied, the instruction is $(i, -1)$. To pour from jug i to jug j, the instruction is (i, j). Instructions should be bracketed by []. So, for example, assume that the four jugs have respective capacities 15, 28, 42, and 44, and we specify that the goal is 35. One possible solution (that should be written into a text file *without* any line feeds or extra white space) is

> [(-1, 0), (-1, 3), (3, 1), (1, -1), (3, 2), (-1, 3), (3, 2), (2, -1), (3, 2), (-1, 3), (0, 2), (3, 2), (2, -1)]

2 Cryptography

Machines take me by surprise with great frequency. - Alan Turing

World War II was very close (and much closer than many people want to believe). On the eve of World War II, the US Army ranked, with reserves counted, 19th among the world's armed forces. This placed the US after Portugal - but ahead of Bulgaria. In short, the Allies did not win WWII with weapons and training, but with brains and information.

During WWII, Turing returned to England from Princeton, where he was studying under Alonzo Church at the Insitute for Advanced Study, to Bletchley Park and into a group called Hut 8. He was the principal designer of the "bombe," an electromechanical device (an advanced version of the Polish machine) for breaking the German ENIGMA machines and was instrumental in breaking other German codes. This was considered by western Supreme Allied Commander Dwight D. Eisenhower to have been "decisive" to the Allied victory. His crypto work was kept a secret until the 1970's; not even his close friends knew about it.

The German machine was a polyalphabetic substitution cipher: it was a *substitution* cipher because one letter was exchanged for another, and it was *polyalphabetic* because the mapping between the letters would change as the message was being encrypted. Had the Germans followed proper policies and practices, this type of code would have been unbreakable at the time (when, please remember, there was no such thing as a computer). Today, the Turing Award is the Nobel Prize of computer science.

This project will give you a chance to test whether you should find yourself a job at the NSA.

- Dictionary: You will be given a listing of words to a language that looks suspiciously like English but is not. The words of this language were chosen by picking letters randomly and deciding that certain letter strings were words. You will be given a file that consists of a dictionary for the entire language.

- Corpus: The corpus will be a large sampling of the "everyday" use of this language (similar to the way we might, for example, collect a thousand newspaper articles and remove all of the punctuation to get a sampling of the everyday use of English words). This large file will consist of a sampling of common word usage: in other words, the frequency that a word appears in the corpus gives you a fairly good idea of the frequency that the word will appear in the encrypted message.

- Message: Dr. Evil has hired an assassin who only speaks this limited version of English. You have intercepted an encrypted message from one to the other and you want to decrypt the message. However, in addition to using the strange language, they are also using a strange encryption method. Dr. Evil and his assassin have met and standardized a two-byte

table; they are using a monoalphabetic substitution cipher, somewhat simpler than the ENIGMA code. They have agreed, for every two characters in the message, that they will instead subsitute two other characters. Note that spaces are also included in this table; the two characters "(space)o" may correspond to "pl" in the encrypted message, and "xv" may correspond to "b(space) ".

The message has the same form as the corpus (i.e. word, then space, then word, then space, then word, etc.) so that you know that the first character in the message is a letter and the last character in the message is a space.

2.1 Input, output

Your goal is, given the following text files: corpus *corpus.txt*, dictionary *dictionary.txt*, and the encrypted message *message.txt*, to produce a mapping *map.txt* that will translate the encrypted file to plaintext. Each line of your mapping file will look, for example, *exactly* like the following:

[f,a]->[d,]

Notice that there are a total of 12 characters on this line (not counting the newline at the end). This will indicate to the decoder that, should it encounter the letter f followed by the letter a, it should translate it to the letter d followed by a space. Note that there may be more than one mapping that produces a valid translation of th encrypted file.

3 Rockets

If we have to have a choice between being dead and pitied, and being alive with a bad image, we'd rather be alive and have the bad image. -Golda Meir

In this assignment, you will attempt to classify objects in a non-academic situation.

Assume that you are the leader of a country. Your country shares a border with another country that is decidedly unfriendly to yours. On a regular basis, rockets are fired from the unfriendly country into yours. These rockets are not perfectly accurate but they routinely cause damage to buildings, interrupt everyday life, and occasionally injure or kill citizens of your country. In short, they need to be dealt with.

Unfortunately, there are numerous distinct hostile elements in the other country and your response as leader of your own country will depend on which is firing the rockets. Of course, there is a great deal of uncertainty as to which group fired which rocket. Luckily, your intelligence services are hard at work and are able to classify a good portion of the rockets.

Your job is, given a list of attributes and classification results, determine which rockets were fired by which group. You will receive as input a text file called *data.txt*. This text file can be opened with any spreadsheet program (e.g. Excel, OpenOffice, etc.). There will be 38,000 records in this data file. The first 35,000 of these records will be classified into one of four hostile groups (namely, 0, 1, 2, or 3) or "unknown" (4). Your job is to classify the final 3,000 rockets and rewrite the data file with these fields filled in.

You may assume the following:

1. Your country has 3 cities that are located at the following coordinates: (10,10), (14,14), (18,10). Notice that these coordinates form a triangle. One of the possible attributes that may be important is which city the rocket targeted. Unfortunately, the only information that you have at your disposal are the coordinates where the rocket landed. The first two fields will be the x and y coordinates respectively of the landing spot of the rocket.

2. The third field will be the political climate at the time of the launch. This is a binary field where a 0 indicates that the climate was relatively calm and a 1 indicates volatility of some kind.

3. The fourth field will be the type of rocket. There are 4 types of rockets, 0 being the most expensive and therefore most accurate and 3 being the least.

4. The fifth field will indicate any modifications made to the rocket. 0 will indicate that no modifications were made. 1 will indicate that ball bearings or other metal objects were packed onto the outside of the rocket (so that the rocket acts like a grenade). 2 indicates that extra explosives were used.

5. The final field indicates which hostile group fired the rocket. The final thousand records will not have this sixth and final field. It is your goal to fill these in so as to maximize the percentage of correctly classified rockets.

3.1 Input, output

The only input will be the text file containing the information. Recall that your output will consist of exactly the same file, but with the final field of the final thousand rockets filled in.

4 Training a neural network

> Using 1000 machines composed of 16 cores each, Google X Lab created an unsupervised self-learning neural network which was fed with pictures extracted from Youtube videos. After training, the neural network managed to categorize images such as human faces or cats with a satisfying percentage of success. In other words, Google Xs neural network was able to create new concepts without any previous data or human intervention during the learning process, which could mean that the artificial neural network was able to generate new knowledge by itself and somehow understand its environment. Of course, this kind of conclusion can sound a bit hasty and ambitious but the results are quite surprising, considering that a normal human brain has around 100 billion neurons for 100 trillion synapses while this network had only 1 million synapses. —Post-cognitive Topics (website)

In this lab, you will train a very simple neural network to learn a very simple *probabilistic* function. A probabilistic function is a function that, on its inputs, has a distribution corresponding to its outputs (rather than a unique input-to-output mapping, as you are probably used to). A probabilistic function will be generated with 3 inputs and 3 outputs from which a large (500 thousand) number of samples will be drawn. Your job will be to design a neural network with 3 inputs, a hidden layer with 3 neurons, and 3 outputs that approximates the given function.

4.1 Input/Output

You will assume that nodes on each layer are numbered so that the input nodes are i_0, i_1, i_2, the hidden nodes are h_0, h_1, h_2, and the output nodes are o_0, o_1, o_2.

Your input will be a sample file consisting of 500,000 samples from the probabilistic function. These samples will be of the following form:

[011,101]

Note that there are exactly 9 characters on this line. The first 3 bits represent the input and the second 3 bits represent the output. The file that you are sent will contain 500,000 lines of this form. It should be noted that you should interpret the low order bit of the input (1 in this case) to be fed into i_0 and the high order bit of the input (0 in this case) to be fed to i_2. Similarly, the low order bit of the output will be expected out of o_0, etc.

Your output will consist of a file that contains 24 floating point numbers, exactly one per line. The first 9 numbers will represent the weights on the edges from the input layer to the hidden layer in the following order:

$$i_0 \to h_0, i_0 \to h_1, i_0 \to h_2, i_1 \to h_0, \ldots, i_2 \to h_1, i_2 \to h_2$$

The following 3 numbers will be the thresholds of the hidden layer neurons in the order h_0, h_1, h_2. The following 9 numbers will represent the weights on the edges from the hidden layer to the output layer in the following order:

$$h_0 \to o_0, h_0 \to o_1, h_0 \to o_2, h_1 \to o_0, \ldots, h_2 \to o_1, h_2 \to o_2$$

The following 3 numbers will be the thresholds of the output layer neurons in the order o_0, o_1, o_2.

Your network will be judged based on the following formula, assuming the sample set is S and w_k represents the output you want from output k:

$$\frac{1}{3|S|} \sum_{s \in S} \sum_{k=0}^{2} (w_k^{(s)} - o_k^{(s)})^2$$

(It may be helpful to you to submit a random neural network at the beginning so that you can make sure that you are evaluating your neural network correctly.)

My grader will evaluate how well your neural network predicts the function. Your grade will be based on how much better you can do than a randomly generated network.

5 Battling the aliens

I don't care if I pass your test, I don't care if I follow your rules. If you can cheat, so can I. I won't let you beat me unfairly - I'll beat you unfairly first. - Ender

Humanity is at war with a race of aliens who are bombarding a certain area of space with bombs. These aliens are predictable in the way they act militarily. The alien high command has picked a random number of soldiers to fire their weapons at specific targets in 3D space. These soldiers fire at different frequencies and with different targets and different weapons. However, you can be guaranteed that all of the alien weaponry functions as three-dimensional Gaussians. Your goal is to make a mathematical model of the alien strategy by determining the number of soldiers that are firing, the probability that a given shot is made by each soldier, and the type of 3D Gaussian their weapon creates.

5.1 Input, output

You will be given an input file with points in 3D space, one per line, and an a priori distribution for how many soldiers your own military suspects the enemy has firing at you. For example, the first three lines in *points.txt* might look as follows. Each point indicates where a particular enemy laser blast has detonated.

```
22.2668358092858200,14.7010913254612450,76.9996826254759800
48.8009335355751760,65.9429970960635100,52.7952528233668100
28.7938985625050850,97.3225835660588800,13.7379856008869470
```

You can expect there to be thousands of such points in your area of space.

The first few lines of the file *distribution.txt* will look as follows.

```
0.00246
0.00253
0.0033
```

You are guaranteed that the numbers in *distribution.txt* will add up to 1. This should be thought of as saying that the probability that there is exactly one enemy firing at you is 0.00246. The probability that there are two enemies firing at you is 0.00253. And so on. There will be 100 lines total in the file.

You job is to return three separate files called *theta.txt*, *mean.txt*, and *covariance.txt*. **If your files are named anything else, your submission will be rejected.** The purpose of the files are to indicate to human High Command the mathematical model of the alien forces that you have determined are arrayed against you.

- The file *theta.txt* will hold lines that indicate the probability that each enemy fires. The number of lines (between 1 and 100 obviously) will be

the number of enemies that you believe are firing at you. The value on line i will indicate the fraction of the total shots that you believe were fired by enemy i. For example, the first few lines of *theta.txt* might look as follows.

0.0062000000000000000000
0.0153500000000000000000
0.0122000000000000000000

This indicates that you believe that 0.0062 of the total shots fired were fired by Enemy # 1, 0.01535 of the total shots were fired by Enemy # 2 and so on. These numbers should theoretically add up to 1.

- The file *mean.txt* will hold lines that indicate the centers of the Gaussians for each enemy. There should be the same number of lines in this file as there were in *theta.txt*. The value on line i will be the point in 3D space that represents the mean of the Gaussian for enemy i. For example, the first few lines of *mean.txt* might look as follows.

  6.028474910011566,3.4885997484013407,80.48710321638431
  28.489020239879586,65.13147361006574,22.61904615159244
  18.39707747591961,70.364258125618,66.7914268339559

  This indicates that $\mu_1 = (6.028474910011566, 3.4885997484013407, 80.48710321638431), \mu_2 = (28.489020239879586, 65.13147361006574, 22.61904615159244)$, and so on.

- The file *covariance.txt* will hold lines that indicate the covariance matrices of the Gaussians for each enemy. There should be exactly three times as many lines in this file as there were in *theta.txt*. The values on line $3i - 2$ will be the first row of the covariance matrix of the Gaussian for enemy i. The values on line $3i - 1$ will the second row of the covariance matrix of the Gaussian matrix for enemy i, and similarly for $3i$. For example, the first few lines of *covariance.txt* might be as follows.

  1,0,0
  0,1,0
  0,0,1
  1,0,0
  0,1,0
  0,0,1

  This would indicate that the covariance matrices for the first two enemies are the identity matrices. (This is highly unlikely to be the case for your example.)

You will be evaluated by computing the information content of the model that your information produces (assuming that you submit a valid model with the correct formatting) and comparing to the results of my simple algorithms.

6 Modern physics

If the Higgs boson is made, it decays before it travels the length of a single proton. It decays far faster than we can observe it. What we observe are the things it decays to. We measure those and run the equations and physics backward to infer that they came from a Higgs boson. -UCLA physics professor Robert Cousins

As you may know, physicists nowadays almost never directly measure something. For example, when physicists were attempting to "discover" the Higgs boson, they were merely attempting to infer its very temporary existence by measuring the byproducts of its decay. The total lifetime of a natural Higgs boson is theorized to be on the order of approximately 10^{-22} seconds.

The purpose of this assignment is to show the kinds of calculations that a standard moderen experiment might require to validate a physical theory. Assume that you are given a system that can be in one of three possible states, a ground state s_0 and one of two excited states s_1 and s_2. For each valid i, over the course of one time step, if the system is in state s_i, the system will emit an alpha particle with probability a_i, a beta particle with probability b_i and nothing at all with probability $1 - a_i - b_i$. Scientists are virtually certain that this particular system transitions between energy states every time step via some Markov process, but it is unknown what that process actually is.

Assume that your experimentalists have set up a method of determining every second whether an alpha particle, beta particle, or nothing was emitted from this closed system. Your job is to determine the physical theory that best describes the results of the experiment. More specifically, your goal is to find the physical theory (i.e. transition matrix) that maximizes the sum of the probabilities of the N most likely state transition sequences that correspond to the results of the experiment; N will be an input to the program.

6.1 Example

Assume that the probability of measuring an α, β, or nothing is given in the following table.

	α	β	nothing
s_0	0.9	0	0.1
s_1	0.1	0.7	0.2
s_2	0	0.3	0.7

Assume that there is a 0.3 probability that the system starts in state s_0, a 0.3 probability that the system starts in state s_1, and a 0.4 probability that it starts in state s_2.

Finally, the detector measures the following sequence of outputs:

"XAXABAXAAXBAXB"

where a X indicates that nothing was measured.

The goal is to determine the transition matrix that maximizes **the sum of** the top, say for example, 5 state transition possibilities. You will need to give me the transition matrix and the 5 corresponding probabilities with their corresponding state transitions. In other words, given the physical model/transition matrix you come up with, you will need to give me the 5 most likely state transition sequences and their corresponding probabilities as well as the matrix itself.

The optimal transition matrix for this problem is roughly

$$\begin{pmatrix} 0.10044974036931184 & 0.0 & 0.8995502596306881 \\ 1.0 & 0.0 & 0.0 \\ 0.5712747299135463 & 0.42872527008645367 & 0.0 \end{pmatrix}$$

6.2 Input,output

The input to the program will include $a_0, a_1, a_2, b_0, b_1, b_2$ as defined above. In addition, you must be given p_0 and p_1, the a priori probabilities that you start in states 0 or 1 respectively. Finally, you will be given the value of N and the name of the output text file.

The output for the example above might yield the following data file:

0.10044974036931184 0.0 0.8995502596306881

1.0 0.0 0.0

0.5712747299135463 0.42872527008645367 0.0

2 0 2 0 2 0 2 0 0 2 1 0 2 1 6.083265724996337E-6

2 0 2 0 2 0 2 1 0 2 1 0 2 1 5.049862212356001E-6

1 0 2 0 2 0 2 0 0 2 1 0 2 1 2.281838969610849E-6

1 0 2 0 2 0 2 1 0 2 1 0 2 1 1.8942082934123542E-6

0 0 2 0 2 0 2 0 0 2 1 0 2 1 1.1460506603099402E-7

Note that, following the transition matrix on the top 3 lines, the 5 most likely sequences followed by their respective probabilities are on the next 5 lines.