SPARK: A Dating Application Utilizing Neural Networks
Final Project Report

Team 2
Christopher D. Miller, Cassandra A. Colón, and Tara Douglass

May 3, 2018

1 Introduction

1.1 Dating Applications

This project explored the area of artificial intelligence, more specifically neural networks, for use in implementing a more intelligent dating application. Our motivation for pursuing this idea came from various research and personal experience. Primarily, we considered the current climate around dating and sought to improve upon the system. Given the desire for constant affirmation, the younger generation, sometimes referred to as The Millennials or Generation Y, tend to seek alternative ways to connect with people around them. This phenomenon has given birth to an explosion of dating applications, both desktop and mobile based, which allow people to connect to others in a variety of different ways. Desktop applications, such as OkCupid, Match, and eHarmony, have come out of the woodwork to give people of various backgrounds assistance in their relationship pursuits. These applications use the matching algorithms which we aim to recreate with our project, yet unfortunately limit themselves to desktop use. The exception is OkCupid which, given that it is free, has the widest pool of people, but mainly advertises itself to a slightly younger population.

Another class of dating applications, including companies such as Tinder, Grindr,

Her, or Bumble, are those which are formatted primarily for mobile devices. These applications are based off of a more superficial process, which includes selecting 'matches' primarily based on how attractive that individual is to the user, since the profiles focus on profile images and have little space for written content or the expression of thoughts, beliefs, and interests. The problems that arise from these applications are the lack of inclusivity and the lack of personalization when it comes to the matching process. These applications are mostly catered to younger individuals interested in less permanent dating solutions, and often lack any sort of algorithmic pairing of said individuals aside from distance, age and gender.

1.2 Roommate Selection

An additional, secondary motivation for our project was the current inefficient and often inaccurate process of roommate matching at Allegheny. From both personal and secondhand experience, we know how miserable it can be to live with someone with whom you're not compatible. Thus, rather than calculating matches by hand, which can be incredibly time consuming and riddled with human error, we hope that our system could also be used to resolve these issues and provide a better roommate experience to our users.

2 The Algorithm

2.1 Architecture

The architecture for this project utilized supervised learning through an actor-critic model. The network works off a actor-critic model because of the large Action space that we had available. An action space is all the possible profiles that the network can choose. And Example of the actor critic network setup can be seen in Figure 2.1.

While the role of the critic network is to give a rating on the person chosen by the network. The critic network then minimizes the cost function of the Q learning

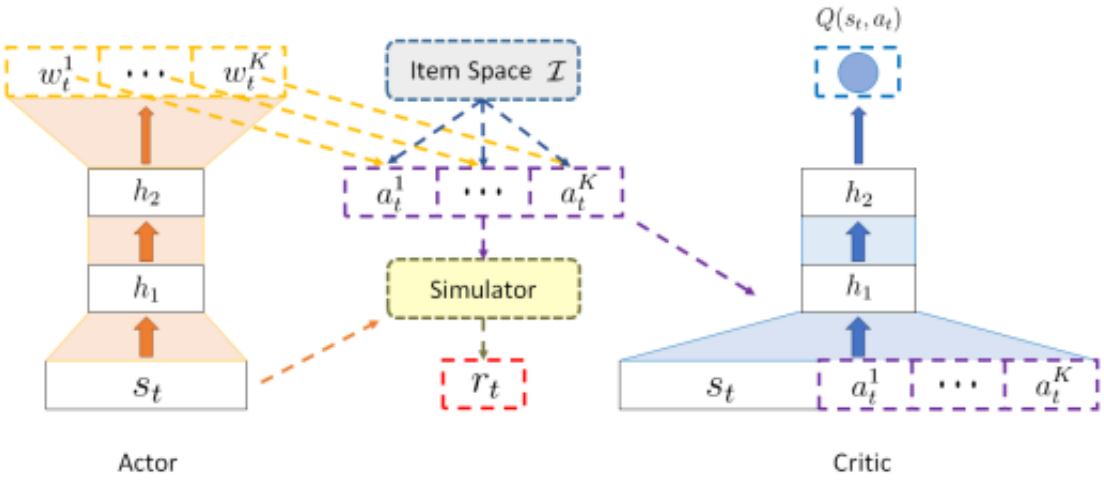

Figure 3: An illustration of the proposed framework with online simulator.

Figure 1: Figure of overall framework, taken from *Deep Reinforcement Learning for List-wise Recommendations* paper by Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Dawei Yin, Yihong Zhao, and Jiliang Tang in 2018

network that is used to implement it. The function the neural network uses as activation is the Sigmoid function $S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$. This takes any input and places it between the numbers 0 and 1.

We began with a population of people based off of our OkCupid dataset. Then we have a selection algorithm that takes the users profile and then goes through the population and picks a profile sends both the attributes of the selected profile and of the user through a neural network with the input nodes represented as the different attributes of both people. The model that we will be implementing is a reinforcement based model. In reinforcement learning you have to define a Markov Decision Process for the problem. Thus defining our Markov Decision Process we need a State, Action, Reward, Transition Probability and Discount factor. For our problem our state is the list of people that the user has given a rating of 6 or higher. The Action of our problem is picking a candidate for our user to rate. The reward is the rating that the user gives for the profile. The Transition Probability is the previous state equals the current state. Thus the algorithm will only find the optimal decision based on reward and all the profiles in the list of people. With all this in

place we are able to use a reinforcement model for use by our supervised learning model. Then the network picks another candidate. This process continues until the network outputs a rating of 80 percent or better, after which it finds all the candidates that have that rating or better and gives to the user.

With regard to algorithmic time complexity, because of the nature of neural networks, there is no set worst case run time. The neural network is continuously running as the user continues to see potential matches and gives them a rating. Therefore the time complexity is dependant on the user of this application. The more frequently they rate different profiles and use the application, the more likely the network is to converge to an optimal solution quickly.

3 Challenges

Throughout this project we ran into various roadblocks along the way. Firstly, feature extraction for this project was probably the most difficult part. Given our large dataset there needed to be a way to express each row as a vector of numbers. The way to get around this issue, which is used in other works, is the n-gram technique. Unfortunately the implementation of this technique would take to long. Thus we had to get creative. The method that we used for feature extraction was taking the numerical data from each row. Such as the age,height and weight.This made it very easy to extract the feature from each profile to run analysis on.

Along with figuring out how to extract particular data points, we also had to take into account the information the different features contained. Some of the answers allowed for a more free form response which lead to difficulty with parsing through the information and assigning it a value with meaning. Additionally, there are 8 essays presented in each profile. However because of the difficulty regarding the natural language processing and sentiment analysis we decided to omit the essays and any other free form responses from the processing of our neural network.

Since this is a reinforcement learning problem and the solution to the ‘problem is left up to opinion. There is no guarantee of convergence when working in this

problem. A user may never find any of the candidates likable thus causing the agent not to converge. A user could have too many variables in their taste and qualities they are seeking that would result in the network to not converge or take to long for the agent to find a truly suitable match.

The hyperparameters of this setup was determining the amount of objects that are being checked every time in the state space, additionally, determining the frequency of choices per epoch for the agent. Also gauging a users interest to see if past experiences will help give some guidance on future ones.

Originally, our team wanted to utilize Neural Evolution Augmentation Topology or NEAT, to generate our pairings within the application. NEAT starts with a randomized generation of neural networks for the given task. Then, after the simulation, a network is assigned a fitness score that is defined by the user, typically based on domain knowledge. The networks with the highest resulting scores are combined to then start a new network, starting the process again with the fittest networks. This process is modeled after Darwinism's survival of the fittest, where the best or strongest networks go on to populate a stronger network.

However, the larger and more important issue with our original method of using neural networks through NEAT stems from the sheer amount of nodes in the network. The dataset we are using from OkCupid has the information for 50,000 profiles. The 50,000 population of people will then be fed into the neural network. When this happens, the attributes for each given person in the population would be split into different nodes. There are 21 different attributes for a singular person, not including the personal essays. This meant that our neural network would have 21 * 50,000 = 1,050,000 for a singular neural network. For NEAT to work effectively, it is optimal to use around 100 networks to produce valid output. This leads to 105,000,000 nodes which is has a computational cost of that is too high to run. The Alden computers would not be able to handle this computational weight. The only way to run something of that nature is to use a Big Data framework such as Twister and have about 10000 different devices to run the computation. A high

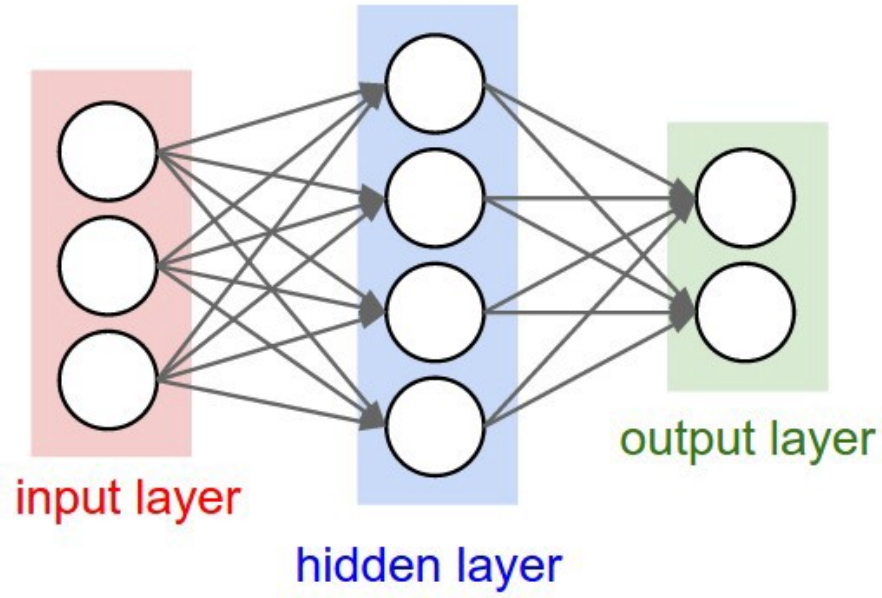

Figure 2: Visualization of NEAT taken from https://medium.com/ml-algorithms/neural-networks-for-decision-boundary-in-python-b243440fb7d1

level visualization of the NEAT framework can be seen in Figure 3.

The roadblocks associated with NEAT forced us to realize the sheer size of the data we are working with. The initial problem arose when we realized that dataset was large enough to crash programs like LibreOffice. There was a considerable amount of work in figuring out how to manipulate this data set and filter it into the person class. This was eventually done though pandas.DataFrame https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.html. The Data frame a two dimensional data structure particularly to view tabular data in the context of python. The panda's Dataframes had more maneuverability when it came to working without dataset or considerably large size.

There was also a short period of time where we began to question the legality of the data we were interested in. The dataset we are using is from publicly available OkCupid profiles. While there was a dataset that was taken without annonomizing the user data, github user rudeboybert https://github.com/rudeboybert collected user data with explicit permission from OkCupid. Each row in our dataset contains: age, 8 personal essays, and health information such as smoking, drinking etc. His github project can be found at https://github.com/rudeboybert/JSE_OkCupid.

4 Conclusion

4.1 What It Does

Currently, our program runs the neural network and outputs various profiles. Each time a profile is outputted, the user must then rate the user. This process will continue and if the user is rating profiles with some of sort of reasoning, the network will slowly begin to output users that cater to the users taste.

4.2 Future Directions

In order for an application as such to be competitive in our current market it cannot be a tool that is only used within a terminal window. The easiest and most obvious way to make an improvement is to create a mobile ready Graphical User Interface for users, just like application like Tinder use now. The neural network would not need to be changed much, it would just work dynamically on the database of current and active users on the application.

Additionally, to ensure a truly functional neural network, there would need to be beta testing period for the network with real people. Utilizing tools within data analytics, user experience surveys and ratings can be compared to the data collected by the neural network to see how effective the network is in finding matches for its users.