Final Year Project

# Declaration

I, Joseph Mckeown, of the Department of Computer Science, University of Reading, confirm that this is my own work and figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person’s work, except where the works of others have been explicitly acknowledged, quoted, and referenced. I understand that if failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly .I give consent to a copy of my report being shared with future students as an exemplar. I give consent for my work to be made available more widely to members of UoR and public with interest in teaching, learning and research.

Joseph Mckeown October 27, 2020

Contents

[Declaration 2](#_Toc68690682)

[Contents 3](#_Toc68690683)

[Abstract 4](#_Toc68690684)

[Introduction 5](#_Toc68690685)

[Background 5](#_Toc68690686)

[Problem statement 5](#_Toc68690687)

[Aims and Objectives 5](#_Toc68690688)

[Solution Approach 5](#_Toc68690689)

[Literature Review 6](#_Toc68690690)

[Recommender systems 7](#_Toc68690691)

[Methodology 9](#_Toc68690692)

[Screens of the Application 10](#_Toc68690693)

[Clothing and Wardrobe System 12](#_Toc68690694)

[The neural network 15](#_Toc68690695)

[Results 19](#_Toc68690696)

[Analysis and Discussion 20](#_Toc68690697)

[Conclusion and Future Works 21](#_Toc68690698)

# Abstract

# Introduction

## Background

This project is an exploration into how neural networks and be implemented in application to improve user satisfaction in the form of an android application where you can get a recommended outfit from a virtual wardrobe. The overall goal it to find how effective a simple neural network is when learning the tastes of a user and how this may lead to the user being more satisfied in the selection made leading to greater user retention. This project uses an application developed in android studio using java with the code being created from the ground up to provide an application which can run on most modern android devices.

## Problem statement

## Aims and Objectives

The aim of this project is to create an android application which will let the user randomise an outfits from their clothes. It aims to include features to improve the user experience like a recommender system, to give more accurate recommendation and connection to a weatherAPI, to factor in the current weather when creating an outfit. It will aim to prove that this implementation will lead to greater user satisfaction and can prove how an implementation of a neural network can learn the tastes of a user over time.

To achieve this an android application needs to be created as the host for the application. This application will need the functionality to be able to receive inputs from the user to create a digital wardrobe from which the neural network will choose the outfits. A neural network will need to be created frolling the multi-layer perception network approach. It will need to be able to learn how an item of clothing if reacted to individually and in relation to other the other clothes in the wardrobe. It will need an intuitive user interface which will allow ease of access to the users making it clear how to approach the application and to traverse it.

## Solution Approach

# Literature Review

Thing to talk about:

* The prominence of recommenser systems and algorithm designed to improve user experience like with social media and Netflix
* The Netflix coemption to create a better algorithm
* The use of reccomender system
* The danger of rellying too much on these kinds of algorithm based learning
* The great implementation of neural network
* The type of neural network and how they can be used.

Using nn for prediction

Metion the history of NN

Creation of perceptrons

Goal with testing is to minimise the error

Perceptron/delta rule

Momentum is used to speed learning up or to avoid the local minima

Look into sigmoidal activation

Mention liniear seperability

Mention the obkect orientated techniche used to make the program more efficient

encapsulation; information hiding; inheritance

mlp can solve non linear problem

layers of neuron in the middle are the hidden layer

can’t use delta rule for the hidden layer as you don’t know the target

werbos developing back propagation

bpg uses a generalised data rule

output deltas propgated back for errors

picton has a book on backpropgation

with BPG error for hidden nodes uses deltas of neuron in the next layer \* the connecting weights

simple adaptive momentum is a new layer for training multi layer perceptrons

adaptive uses

relevant under and over fitting issues considered

too many hidden layer nodes can lead to over fitting

one layer works for any continuous function

2 or more is fore deep learning

Might want to consider using a roullete wheel approach

https://brilliant.org/wiki/backpropagation/

## Recommender systems

This is a system which is a subclass of information filtering which predicts the preference of the user. These are usually things the user wants and the goal is to be able to predict these from the users habits. “RS are used primarily for individuals who lack sufficient personal experience or competence to evaluate all the choices.”

Non personal rs recommendation are typically not associated with standard rs research. The system is defined by the users preferences and constraints. The data to collect about a user can be explicit like their reviews or they can be interpreted from the users action. This can be like a the about of retention a user has on a certain site or video. So for my recommender system I need to be able to define the type of data I will collect about the user and how I will interpret that data. To do this I will create some case studies which a predictable input which should provide a mostly predictable output form the app.

The importance of recommendation systems is the amount of influence recommendation have on everyday life and routines. It streamlines the decision process and reduces the amount of choice having to be made. This can help by reducing decision fatigue. This being caused by the overwhelming about of choice which is facilitated.

It’s also improved as it often compares other users of similar taste pattern to find items which could be recommended to them. This process is called collaborative-filtering

These recommendation become more and more important in modern life as due to increased ease of access to information and items though the internet, we are faced with more and more decisions each day.

The role of a rs is to benefit the user and the owner. For the user it’s the ability to have a better selection of choices while the owner benefits as it often mean greater profit. This can be due to the user finding something they want to buy or the customer experience being improved leading to longer use of the application/site and establishing consumer loyalty.

Key point into why the service provider would want to implement a rs system would be:

* Increasing the number of items sold
* Sell more diverse items
* Increase the user satisfaction
* Increase user fidelity
* Better understand what the users want

An RS needs to be able to balance how much it benefits the user and the owner. It needs to be able to cover multiple bases at the same time.

The main focus of my recommender will be to “find some of the good items” which are provided to the app by the user. It will use this limited data set to provide the best options.

Along with this it will have to be able to “recommend a bundle” of items as the core of the app is to be able to create an outfit, so the synergy of items will play an important part in the recommendation process.

It also will use recommendations to “improve the profile” of each user using the app as it will take the input taken through the app in it’s accept or rejection of outfits to build a stronger profile.

<https://en.wikipedia.org/wiki/Recommender_system>

Francesco Ricci and Lior Rokach and Bracha Shapira, [Introduction to Recommender Systems Handbook](http://www.inf.unibz.it/~ricci/papers/intro-rec-sys-handbook.pdf), Recommender Systems Handbook, Springer, 2011, pp. 1-35

<https://link.springer.com/book/10.1007/978-0-387-85820-3#about>

https://www.healthline.com/health/decision-fatigue

# Methodology

<talk about why you choose andoird stuio and java as the programming language. It would also help by showing information like how many device use android in the modern day. Esssentiall prove why you are using it. >

The design of the app would need to be able to facilitate the main function of the app. These functions being:

* Neural network recommendation
* Wardrobe management
* Clothes additions / deletions
* Outfit storage

To accommodate this, the app can have separate screens to focus on each task. With this idea, 3 screen would need to be created. It would also need a central screen which would start the app and would be where the user would spend most their time. For this the recommendation makes the most sense as the main screen as this it the key functionality of this application.

## Screens of the Application

### Screen1

For this main screen the user would need to be able to request a recommendation from the current items of clothing available and to present it on screen. Along with this it also need the functionality for the user to decide if they like or dislike the recommended outfit. This can be done through a having button either side of the recommend button. These would provide each option respectively. As well, due to it being the main screen of the application it would need to have suitable menu functionality. This can be done by having buttons at each corner at the top of the screen. These can lead to both the clothing item addition screen and the wardrobe management screen. This would follow standard designs where the options and extra feature are accessible in other screens.

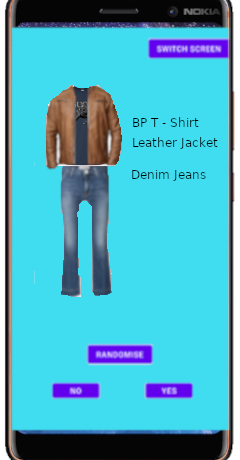


Figure 1 Early Demo of main Screen

In the main screen the main process will need to be the recommendations. When the user is using this screen, they should follow a general cycle of interactions. This would be them obtaining random outfits and deciding whether they like the outfits or not with the occasionally adding cloths to the application. From this the design for the usage of the main screen can made. In this it also shows the basic flow of the recommender function as well. It’s structure is that once a request is made to get a recommendation it will find the outfit with the best fitness at that point and the present it for the user to judge.

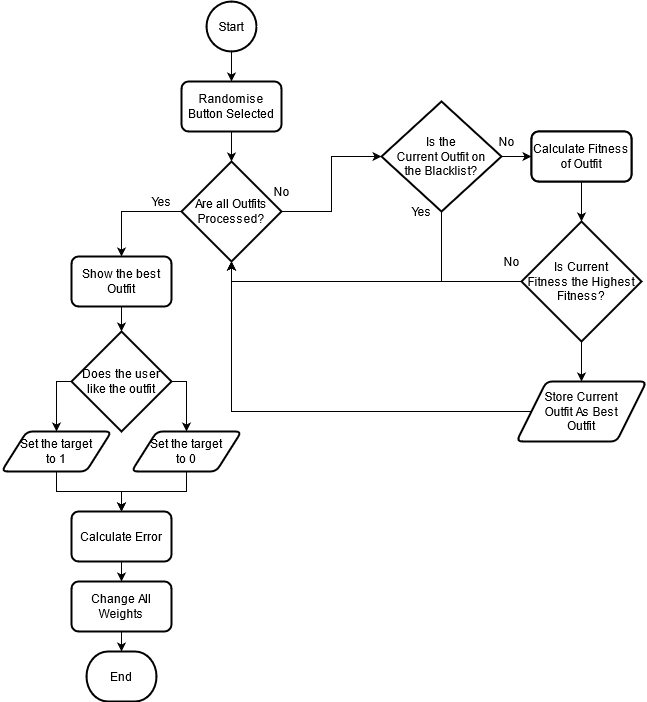


Figure 2 Flowchart of the main Screens main process

### Screen2

The next screen of importance is the wardrobe screen. in this the user will be able to see the entirety of the clothes which they have entered into the application. It will allow the user to browse through each item and highlight them to gain more information about the item of clothing. This can be information like a score based on it’s current fitness. These kinds of feature can help the users better understand how the application is functioning and can also allow help them with highlighting items of clothing which they prefer. This will help towards the goal of a better user experience as it can help them better understand their tastes through this app along with potential highlighting patterns of decision which the user was not aware of previously.

Another necessary feature in this screen is to edit the properties of the item which are in the wardrobe. The user should be able to change the majority of the key information about an object like the name, image representing it , delating the item and even what category it belongs in. As this I changing key information about the item’s the NN is based off some consideration have to be made. For both the Name and the Images they should be able to be delt with within a Clothes class. This is because they are the properties to define the clothes which are not intertwined with the NN. However If the user would want to delete or to change which sub category the item belonged in accommodation would need to be made to ensure the NN says accurate. This would be ensuring the weights are removed accordingly with the deletion of an item along with removing any outfit which may have contained the item. For the changing which type it belongs to, it would need to have the have all the outfit weights containing that item removed as those outfits would not longer be possible. Along with that it would need to create whole new outfit weights for all the new outfits this type change would have created.

##mock up here

### Screen3

Another screen of the application is the Item addition screen. Here would be the main hub for the users to be able to add the items of their wardrobe into the application virtual wardrobe to be created. This screen would need to have the option for the user to input all the relevant information about each item of clothing. This would will probably be the name of the item, a picture of the item and the sub-type which the item fits under. With all of this information it should be ready to add a new item to the virtual wardrobe and the NN.

To add new item to the NN it would need to consider how it would create the new weights for it. individually, all it would need to set up the creation of a new weight. When it come to outfits, all the new outfit created due to the inclusion of this new item would need to be indexed. Along with this the weights would need to be created for the respective outfits which are created. this may take a long time as the amount new outfits which need be made with each new item added is generally exponential##CHECK THIS# meaning that with a larger wardrobe it may take some time to calculate all the outfits. This could be resolved by either implementing multi-threading when it is possible on a device or to have an item limit where only a certain about of items can be added where the outfit limit would not be exceeded.

#mock up here

This screen would be implemented through creating a separate activity within the app. Within this activity, it contains the main features mention, being the ability to choose our options for a new item of clothing.

On the screen, the user is provided multiple options:

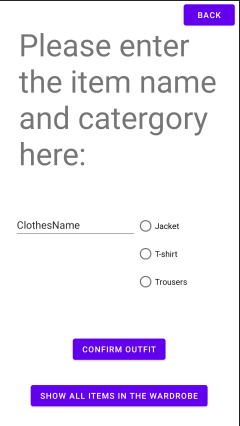


Figure The UI of the item adding activity

* 3 buttons which align with the 3 main categories for the clothes in the clothing classifier.
* A button to show all the items which are currently within the wardrobe
* Textbox for the user to input the name of the item of clothing
* Confirmation button which creates a new item of clothing in the wardrobe

With these option the user is able to engage with adding item to the app. Some consideration need to be made when implementing as to ensure that an invalid item is not entered into the wardrobe, the option are checked on confirmation of the Item. This require that the would be clothing item to be entered is validated. This means it has to be checked to ensure it has the right attributes being a name and a type. Once validated it will be added to wardrobe. If the requirements for an item clothing in the app was expanded the validation would equally need to be extended. To prevent the app from crashing a try catch statement was used. This would TRY adding the item of clothing into the wardrobe and if any part of the process would error it would stop the process and display a message on the main screen telling the user to choose valid inputs. Doing this prevents the app from crashing from invalid input while also making the user aware of the issue.

Another consideration is how to send the data on the wardrobe and clothes between the activities. This is because when you switch between screens the app closes the previous screen meaning the key information needs to be passed around each activity. This can be done by sending intents across. Intents are used to communicate between multiple components in an application. By using intents, we can send the wardrobe information between each activity. This means that each screen needs to accommodate from packing the data into intents to send over and also to receive and unpack the intents. This can be done through be sending the data as multiple variable arrays or taking the object and making it a passable/bundle.

## Clothing and Wardrobe System

The main objects within the application are the clothes. There needs to be a way to represent the clothes of a user within the app. To do this it would need multiple parameters to define what the item of clothing was. This could be done be categorising each item into a general theme of similar items. This would make it easier to be able to create outfits as there could be easily defined parameters to create an outfit. To try and solve this problem, the idea of a virtual wardrobe is created. this is where the all the representations of the items of clothing from the user are stored in a particular order and method in a virtual wardrobe object. This would help aid in the classification of the item for creating an outfit as it an outfit would need to pick the required items of clothing from each section of the virtual wardrobe.

To achieve this each item of clothing in the application would need to be classified as a type. The goal of which is to prevent item which would be incompatible in real life E.G. not wear 3 pairs of jeans. A way in which this can be executed is by having each of the clothes to be sub-dived into 3 main categories being a Top, Under Top, Bottom. These would represent the basic components of what would make a usual outfit. This can be expanded so that the minimum which an outfit could comprise of is a under top and a bottom clothing item as generally this is the least which is considered acceptable. By using this clothing classification, it makes it simple and effective to classify clothes and prevent conflicts when creating outfits. It can also allow for further customisation with the user as you could change the minimum and maximum requirements needed for an outfit to be considered. Another way in which it can be expand in is to include more sub-categories. This could be very useful as it gives more specific option for the users to classify their item which will help track their preference in certain subtype of clothes. This can help provide more accurate results as it the NN can have the influence of preference in certain subtype play a heavier role it which it selects as the most appropriate outfit. This is because the more specific the categories the easier it become to implement category based filtering/learning. This because the more specific the categories are, the less generalisation have to be made if implementing it into the learning process. For example If the user likes jeans but they are classified under long bottoms(E.g. jeans, trousers, leggings) it will recommend others within that sub category even if the user has no affinity for clothe of that type.

A benefit for more defined sub-types it can allow for more implementation of more traditional recommender system methods to be introduced like a collaboration filter. If the application was used by multiple users and the data was taken and stored, then a collaboration filter could be created. This could compare other users on the application who have similar tastes or trends in their clothes and outfits to try and recommend other thing which those similar users like. This makes it much easier to present recommendation when there is little data on a user as it can pull from users with perceived similar tastes. However, this would need a large data pool size to effective. This is both because of taste in clothing very subjective there would be cases where is the pool is not large enough more niches taste within the system would benefit less from the collaborative filtering and could lead to potentially longer times to learn the users tastes.

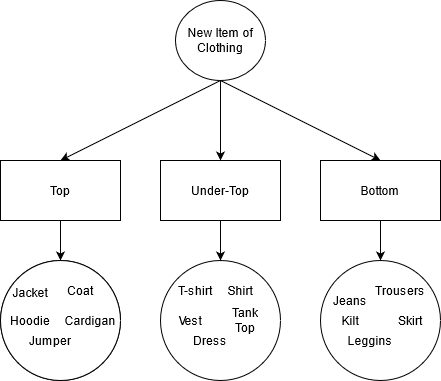


Figure 3 The Design of the Clothes Classification System

The implementation of this was to create a class for the items of clothing. Within this class, each object would contain the name, id, type of item which it is. This would allow for every item of clothing in the app to have a standardised format of information which makes it easier to create and access information about the item of clothing. These objects would be created when ever a new item was created within the app and then would be stored within a virtual wardrobe.

To represent the virtual wardrobe, it would need to be able to stores each individual item of clothing added to the app along with being able to be changed through addition, deletion and edit whenever the user so chooses. application. The way this can be implemented is through creating a multidimensional ArrayList. This would allow for clothing items to be moved on and off the list with ease. The multiple dimensions of the ArrayList are to accommodate for the number of types which are used in the clothing classification system. This is because each dimension represents an unique array list containing all of the item of clothing within that type so for 3 types there would be 3 dimensions respectively.

This is done to create to help to ease the process of creating an outfit. This is because the criteria can be set to one of each item from each dimension of the ArrayList to create a new outfit. This format helps to properly define what is considered an outfit in a wardrobe. By defining it, all the possible outfit from the wardrobe can be easily created through by creating outfits from 1 item from a category and all other items from the other 2 categories. Along with this it would affect any sub-types which may be introduced as they would still be categorised under these main three categories.

## The neural network

A model needed to be decied upon for the neural network as it is the core compenent to this application. The inputs and structure of the NN need to be established. For the inputs it takes the inputs from the current item of clothing/outfit being rated. So when let’s say clothes item has been chosen to find it’s value it will input a 1 while any other piece of clothing within the category will produce a 0 input. Each input has it’s own wiegth attached to it which will changed depending on the results from the users. Once the inputs have been calculated, this will produced a input for the hidden layer to the output WHICH I NEED TO MAKE FIRST. EXPLAIN WHY HIDDEN LAYER IS GOOD. The network is not fully connected as there are two sets of inputs. One for the clothes and the outfits being a combination of clothes. The amount of outfits is the amount of items of clothes in each category multiplied together so when there are 2 of each category there will 2\*2\*2 meaning there will be 8 possible outfits created so 8 possible inputs. The goal of separating the clothes inputs an the outfit inputs it to allow for the NN to learn how certain piece of clothes may not go well individually but may work really well in niche situation. Along with this it has the goal to learn wether taste in individual vlothes is more important or the synergu between the item is more A picture containing text, electronics

Description automatically generatedimportant.

The problem trying to be solved is also not linearly seperable. Linear seperbily problem are those who can be can be defined by a straing line on if the it will be one result or not. This is better benefiting classification types of model with NN be can be considered within a predictive NN.

This type of NN is not using a data set and will learn unsupervised. Unsupervised learning is when it learn information without any tags attached to then. The described the application well as the algorithm will not know which kinds of outfits will be accepted or rejected by the user. This is due to the nature of clothing and how it is contanly changing in the tastes of the user. This is why the algorithm needs to be able to adapt quickly to new item being added to the system.

For the NN to be able to learn it will follow this structure. Once an outfit is requested by the user it will runs all the possible outfits through the NN. It will output the results for each outfit which will be compared to the other outputs of the other outfits. This can be considered the fitness of the outfit taking inspiration from some methods used within genetic algorithm. In GA fitness is computed on each generation and used to find which chromosomes in the population are best fit to reproduce and create the next generation. This is implemented in this MLP NN by having the having the fitness of the outfit compete against eachother. The selction process could be done in multiple ways such a roulette, rank or (FInd Another). The method I choose was rank selection for the outfit where the outfit which produced the highest result in the nueral network would be chosen to be presented to the user.

In a supervised nueral network, once the result was computed it would be compared to the target result and the change appropreitely. Unlike that in my unsupervided network the target is decided by the user. This is dictated by wether or not the user presses either of the accept or reject button within the GUI. This will set the target for the either 1 or 0 respecitvly. Once the target had be decie by the user, the NN will there change the relevant weight relating to the result. This will create the result where items of clothing which are less desired will show up a lower rate.

An issues which is procuded within this kind of system is that if only the best outfit is chosen by the network each time, the same outfit will be presented by the system on each run if the outfit was always accepted. To alliveate this a solution was introduced where once an outfit had been chosen it would be put onto a blacklist. This blacklist would exclude a chosen outfit from participating in the rank selction whilst on the blacklist. This work in a similar maner in which dataing site will not not repeat a selection once a decision has been made. However unlike those where the goal is to show as many people as possible, the user may want to use the same outfit again. This is why a timer isintroduced to the blacklist where, when the amount of clothes blacklisted exceed a portion of the overall wardrobe size it will add a random outfit back into the selection. This can also work well in the basis that an individual clothing item’s weighting may become higher or lower will the outfit’s would have changed little allowing for outfits which are like to be more likely to show up even after a an item of clothing has a poorperforame in other outfits.

Another way in which this issue can be overcome it through the other selection method of roulette as it this method gives a proportial chance of an outfit being chosen based on the fitness prodced from the NN. This means that outfits which score well are more likely to been chosen but still allow for other outfits to be chosen as well.

Problems (tasks) descriptionsAlgorithms/tools/technologies/etc. descriptionsImplementationsExperiments design and setup

the neural network has to

* Have a interatable ui with multiple screen/ acvtivies which the users can move through
* Creating an algorithm to predict the outfits
* A system to implement the clothes
* Ways to move the data through different screens
* Learning the problem
* Creating test cases to see how effective it is
* Preventing the best outfit from always being chosen or shown.
* Potential implementation of weather api
* Ability to store the information in the app form later usage
* Implementation of backpropgation
* Theory of how to implement a reccomender system and how that could implove the initial learning and the theory of online connectivity.
* How potentially implimnetaion of genetic algorithm can also be introduced to improve the performance as goal to find the fittest
* How the system was made to be ever evlovling with more items added.

# Results

# Analysis and Discussion

# Conclusion and Future Works