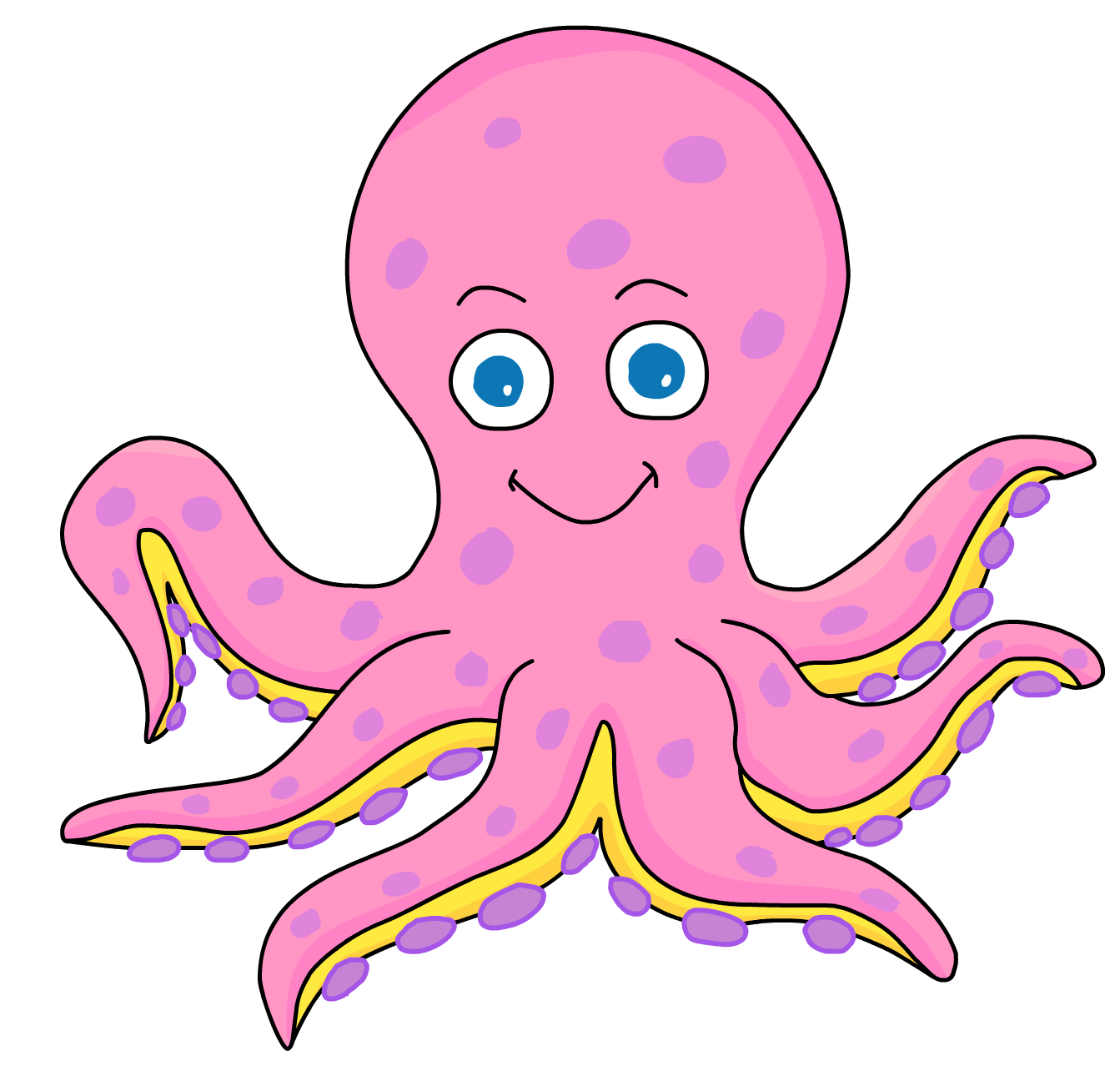
# G54GPP

# Interim Report

# *Literacy games for Key Stage 1 children with dynamic difficulty for individuals produced by machine learning techniques*

****



## **Oliver Bodinar, Lawrence Cate, Robert Sadler**

# Introduction

Literacy education is one of the formative parts of primary school education, but many pupils underperform in this area. Tackling this issue and finding a way of boosting literacy could have a dramatic impact on pupils’ overall education. The traditional methods of teaching will always have a role to play but schools and the wider education community should think about the new methods made possible by developments in IT. Schools have been slow to adapt because they need to be confident in the effectiveness of teaching methods and they may require additional staff training. So despite the availability of new classroom technology resources, many teachers do continue with traditional methods (“Slate evolved to paper and paper evolved to tablets”).

We believe educational games could be better designed and deployed so teachers can deliver education in a smarter way. Computer games have long featured in classrooms, but are seen as a fun distraction rather than a part of core curriculum delivery. Looking at the existing educational games, BBC Bitesize [ref] is the most common platform within the British school system. BBC Bitesize games are very specific, and are not suitable to be played multiple times because they follow a fixed structure and are only aimed to be played for a single session. Other commercial products such as KidsSpell [ref], Purple Mash [ref], similarly offer a similar format of but with a larger variety for games. The former however is based upon Adobe Flash Player, which is soon to become an unsupported plugin (Adobe, 2017), as well as being incompatible with IOS devices which have an 75.64% share of the tablets market (StatCounter, 2016/7). Schools are increasingly acquiring iPad’s in particular for educational purposes. KidsSpell also includes adverts which are inappropriate for children (KidsSpell, 2017).

Our solution is a suite of games themed under the name ‘WordZoo’. The games will initially focus on reading comprehension, spelling, and phonetics but this scope could be expanded later. The games will align with the national literacy curriculum. The games themselves are based on an existing project ‘Savannah School’ developed for G52GRP which focussed on the design and HCI aspect of games for young children. We will implement the findings of that project in our software deliverables. The WordZoo system will track pupils performance across all games, and will intelligently choose which words or concepts to present to pupils. The system will also present a teacher dashboard which will allow them to evaluate individual pupils’ performance, and give them feedback which allows them to intervene.

# Project Specification

Full project specification can be found within the Appendix

# Motivation

Education is a critical task for creating a productive workforce and society, and the UK despite having one of the best resourced and developed systems still leaves some pupils under attaining. The 2017 Department of Education report [ref] identified that 24% of primary school pupils are below the expected level for reading and writing respectively. This means are significant number are falling behind. It is also important to think about the 25% who achieve above the expected level, and may not receive teaching to encourage their progress whilst the teachers focus on getting the rest of the class to ‘working at expected standard’. This ‘one-size fits all’ [ref] approach to education does not get the best out of every pupil.

We hope to create a system that can boost the attainment of all pupils, by adjusting to their individual level and provided a personalised experience. If deployed in a classroom, this could allow teachers to focus on pupils who are below the standard. We have chosen to focus on the Primary School KS1 age group because of the low complexity of information to be delivered, and this requires simpler games and data compared to targeting GCSE education for example. However, the conclusions and methods developed during this project could definitely be reapplied in other educational contexts.

Figure 1

Figure 2

# Aims and Objectives

WordZoo aims to:

* Evaluate how educational video games can boost attainment in young children
  + The games will aid 5-7 year old children with their literacy education
  + The games should be entertaining and user friendly
  + The games will be tested by evaluating pupil performance in the game and by interviewing the teachers involved in the software testing
  + The system will provide detailed feedback to teachers about how pupils are performing
* Evaluate if personalising the games to individual pupils’ performance via the application of machine learning to performance data has a significant effect on knowledge retention
  + The games will be A/B trialled with and without the personalisation component

# Related Work

There is already a lot of academic concern for how to deploy technology in education and respond to the new hardware and software capabilities. Mark Schneiderman, director of education policy for the Software and Information Industry Association said “the factory model that we’ve used to meet the needs of the average student in a mass production way for years is no longer meeting the needs of each student.” meaning that pupils are not responding effectively to centralised classroom learning, and need an educational experience that is not necessarily more entertaining or technology-centric but one that better develops the advanced skills now needed for the modern workplace where many repetitive jobs are themselves being taken on by robots and advances in machine learning/AI.

In Game-Based Learning: Challenges and Opportunities Pauline Rooney highlights that the current generation of ‘digital natives’ are “fundamentally different from previous generations of students because they have spent their lives immersed in digital technologies … such characterisations have led to claims that implementing innovative, student-centred pedagogies supported by e-learning technologies, such as serious games, is imperative because current students no longer respond to traditional instruction.” In practical terms this could be achieved via the development of advanced educational games.

There is currently a lot of related work into ‘customised’ or ‘personalised’ learning. Zoran Popovic who developed problem solving game Foldit states “You can also try to figure out what are the optimal pathways to conceptual understanding for every kid, based on their preference of learning”. [reference] “Student customization represents an important advance because it recognizes that pupils come from different backgrounds, interests, learning styles, and ability levels.” Personalised learning could be achieved using software that can match it’s pace to that of the user.

There is very little research into applying machine learning specifically in educational games, and we hope our project can explore the hypothesis that machine learning can be used within games to increase the learning speed.

In ‘Digital schools: how technology can transform education’ (2012) Darrell West highlights that “Video games and augmented reality represent ways to engage students and teach them important skills and concepts” - and later that “Game-based methods may benefit the poorest-performing students most”. Prensky says “Today’s students are no longer the people our educational system was designed to teach”.

Unfortunately there is limited data on young children's’ educational history. Schools mainly track data on assessments and attendance [reference could we get this from interview?] .

# Methodology

The chosen methodology for this project was SCRUM. It was important that we were able to take an agile approach to change as it would likely occur. Scrum involves weekly or bi-weekly ‘sprints’ with specific deliverables and would allow us to rapidly prototype and make many improved iterations. The initial plan was to have each group member contribute to all areas of the project, however this was quickly dismissed as it resulted in confusion over ownership, responsibility and direction of each area. In practice Agile development real world limitations, such as when applied within a University setting where outside projects will divert attention away from this.

The workload was then divided into three main tasks Web and Database, Client Side Games including their Server Interaction and Machine Learning. One team member would be assigned to each area to oversee progress and lead it’s direction. Members are able to contribute to other areas, but their individual focus would be on their assigned area. Lawrie would head the Web, Database and Report Writing, Oliver would lead Client Side Games, Server Interaction and Contact with Schools whilst Rob would focus on Machine Learning, Graphics and overall Project Management.

We had planned to create our own neural network, however after considering the time and work required it was decided to use TensorFlow, as it’s the industry standard for Machine Learning <Reference> and would provide the best platform to generate the most effective results. The games would be developed in Phaser, an open source HTML5 Javascript game framework, and developed within the dedicated IDE, Phaser Editor. Phaser Editor incorporates and supports JavaScript development, matching our existing knowledge of Java. Node.js will be used for server side work. It was chosen because it is the fastest commonly available solution, would integrate the best with JavaScript from the games and allow the most effective use of sockets and input connection.

# Design

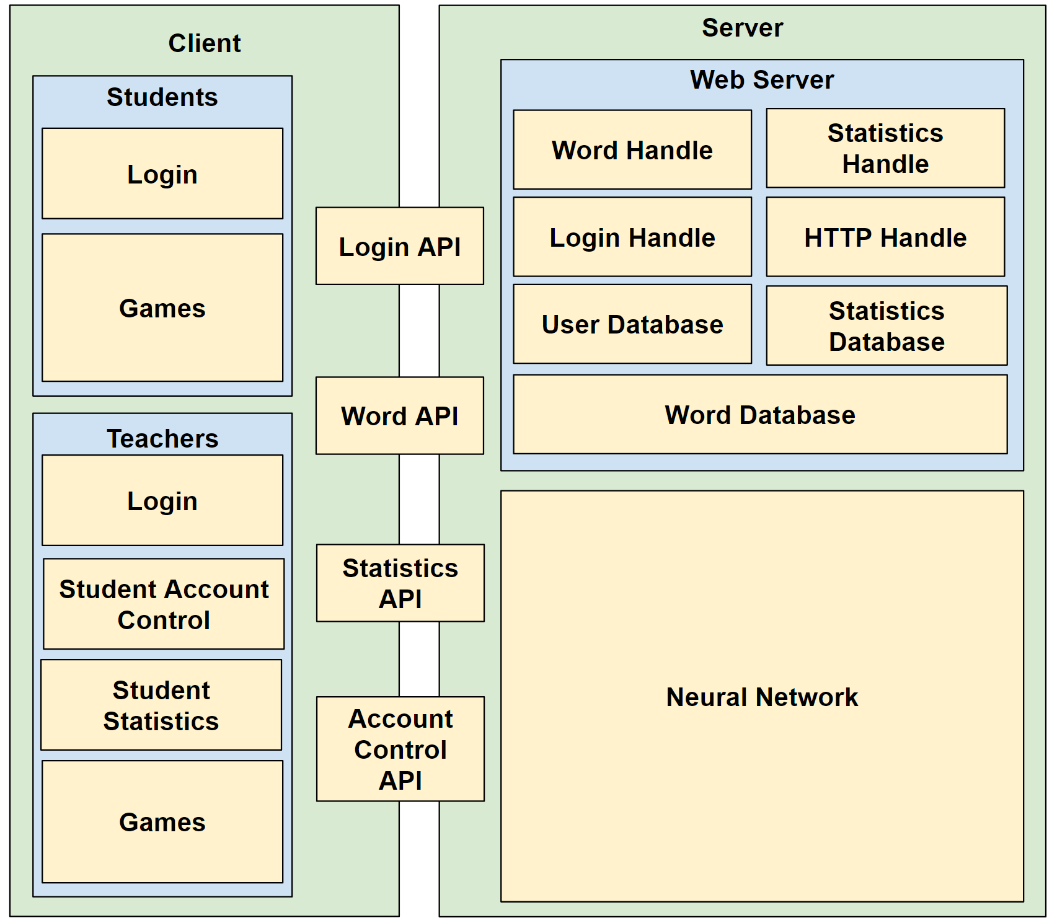
**Overall**

Figure 3: Block Diagram of the Overall System

Figure 3 depicts the components of the overall system. The design is divided in two sections, client and server, with APIs in the middle connecting the partitions. The size of components is an estimate of how much of the overall system that component will be. The neural network element will be the largest component, and will therefore require the most resources and time to create.

The flow will be simple. Each game will take in words provided by the server, and will output data for us to analysis. The data outputted will be for overall analysis, training the neural network, or HCI evaluation of how good the games are.

**Educational Games**

The games will be built on top of the previous work in Savannah School [XXX]. Taking the feedback from the previous trials, the games will be updated and redesigned to incorporate the feedback. These games will be built within JavaScript using the Phaser Library [XXX] to make sure they are cross platform compatible.

The main design philosophies are:

1. The games will use an animal mascot.
2. The games will incorporate bright colours and animations.
3. The Game Over Screens must always have a positive message to them.
4. Interactive instruction screens with limited written instructions
5. Lots of use of background detail
6. Touch controls must be compatible
7. Games must have a deterministic end. Avoid “infinite” game time, and aiming to have an average play time of one minute.

Images are of the game designs, and their developed final products are available in the appendix.

**Teacher Feedback**

Once again, building on top of previous feedback, the presentation of data on students for the teachers benefit should be simple. Simple charts to plot progress over time, and easy to understand explanations of the data.

For example, if a child does better in one type of spelling games as opposed to another, it can be suggested that their style of learning favours one approach. This is information and a teacher or a TA could use to assist them in learning a literacy concept.

**Machine Learning**

The machine Learning section of the project can be broken down into three areas, Word Analysis, Simulation and Selection.

**Word Analysis**

In order to judge the difficulty of a word, a network will be designed to assign a value indicating its’ difficulty. The reason for using a network over manually inputting data is that all reading lists focus on concepts as opposed to individual words. The curriculum is broken down into sections, focusing on phonic patterns, and then clusters words upon that. However, within the group, it is hard to distinguish which words are harder. For example, the words “jump” and “trumpet” both have the same “ump” phonic within them, and so would be considered by the curriculum to be of the same difficulty. However, it is obvious to tell that “trumpet” is more difficult due to its’ increased length. There are multiple factors apart from certain phonics patterns that affect the difficulty of the word.

Therefore, a simple website was created with two buttons, each with a randomly selected word from the curriculum. This website was to be used by ourselves, and teachers to build up a training set to train the Word Analysis network on. The user was asked to press the button for the word they believed to be harder. Each word was randomly generated.

Once a large enough training set was created, we would then use this to train a network which, when given two words, would be able to output a value deciding which word of the two was harder. With the network reliably trained, we would then use it within a Bubble Sort[REF] function to sort the words from the easiest to the hardest word. Every word is compared against every other word, and the Neural Network decides if they are in the correct order, or require swapping. With the words sorted in order, the position within the list would signify it’s difficulty.

**Simulation**

This network will be trained with real data from the children in user trials. Whenever a child completes a game, the following data will be collected:

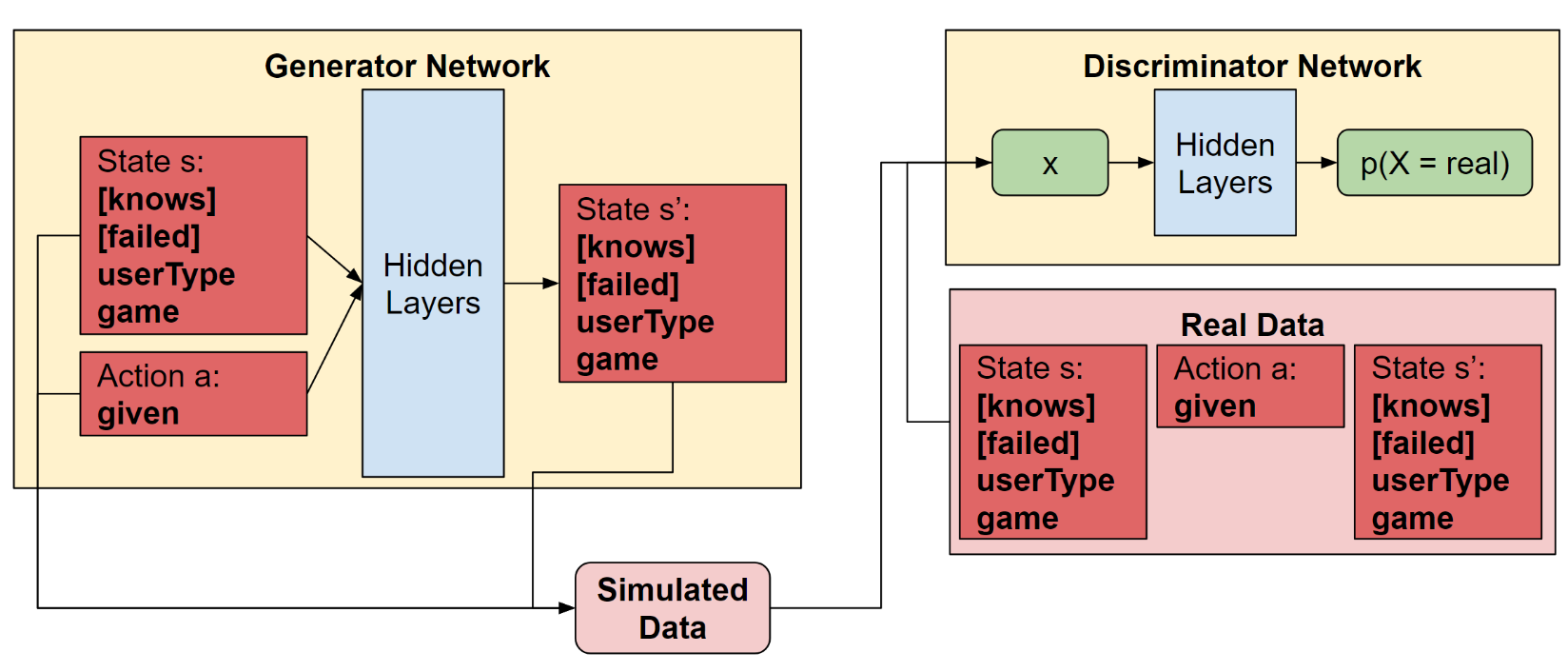
When a child logs into the system for the first time, they will take a small quiz to help identify the type of user they are. We will then use clustering techniques to cluster the data, and assign a userType. We believe this is important to provide the individualised experience we are aiming for. The more clusters we have, the more specialised the experience, but this may introduce noise for this network. We will need to experience in what gets the best performance.

We also record what game they are playing, and then five lists. knownList and failedList are representations of what the child already has seen. The knownList indicates what they got correct, and the failedList indicates what they got incorrect. The givenList are the list of ten words that they were given to play the game (gameName) with. Once they have completed the game, the newKnownList and newFailedList indicated what they now know and what they have now failed on. Effectively, we are tracking if we give them ten words, are they failing or succeeding with those ten words.

This data will be used to train a Generative Adversarial Network (GAN). The networks has two neural networks, a Generator and a Discriminator. The Generator is trying to model the real world data as accurate as possible in order to try and fool the discriminator. The discriminator is trying to decide if the data given is simulated or genuine. The “arms race” approach of both networks working against each other will provide a good training model to reliably simulate how a child would react when given a list of words. This method has been used in Computer Vision [GOOGLE REFERENCE], and we believe that it can be used for finding patterns within this data.

The discriminator is optimised to increase the likelihood of giving a higher probability to real data, and the reverse for generated data. The Generator is optimised to increase the likelihood of the generated data being given a higher probability.

These two networks take it in turn to train each other with the following formulas. Over time, these will gradually converge where the Simulator network will provide realistic and accurate data based on what we have collected.

Another added benefit of using this approach is that we will be able to generate more data for training the Selection network if we are unable to collect enough substantial data from children trials.

**Selection**

Figure 4 - Overview of GAN Network

The other two networks are complementary components to the Selection Network, which is our main network. This network is responsible for picking ten words to give a child based on what they currently know. We will use Q-Learning [REF], a type of reinforcement learning to achieve this.

Q learning was most famously used by Google Deep Mind [REF] in their Atari example. Effectively, what we are trying to produce is an Artificial Intelligence system that is playing a game to maximise what the child learns.

Q-Learning makes use of the following:

**State** - A state is a represented on what the network is looking at to make its’ decision.

**Action** - An action is a decision that the network makes to change the state (or not) into another state.

**Reward** - The reward is a value that shows the state is a progression towards a goal, or a value of a state. Q learning is trying to maximise its rewards.

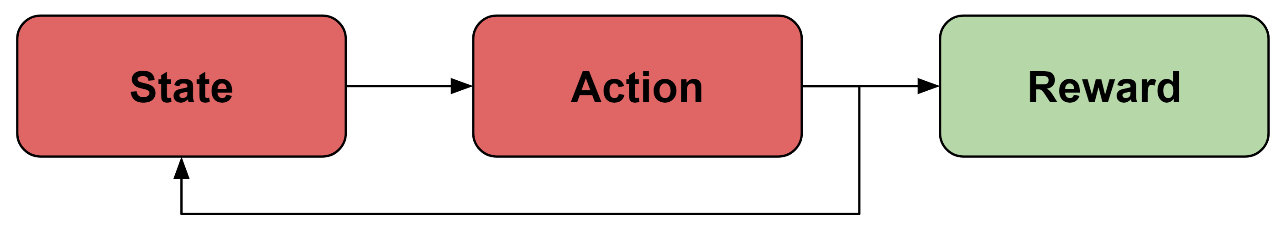
**Gym** - Gym is an environment to generate states based on action inputs. In the Atari Example, the Gym was an Atari Emulator.

Figure 5 - Flow Diagram of reinforcement learning

In our system, our States are consist of : [knownWords] | [failedWords] | userType | game. Our set of actions is the list of all words in the system, and an individual action is one of these words. The reward value for each state is calculated as:

We make use of the data from the word analysis network to assign the difficulty value to each words in the two lists. We do this so there is a greater value on more difficulty words, instead of the network giving the child lots of simple words which they easily know. This will not help them advance their learning. We want to maximise the difficulty of words that they know, whilst minimising what they are failing at.

Our Gym will be the Simulation Network. We will be able to model what would happen when given a state and an action. We will only use the Generator portion of the network.

Q Learning works by selecting an action that it believes will result in the greatest reward. It focuses on short term, but can also be affected by long term rewards.

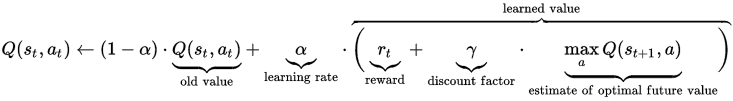


Figure 6 - Q-Learning Formula Source: XXXX

Q(st,at) is the value a state is given when a certain action is given to it.

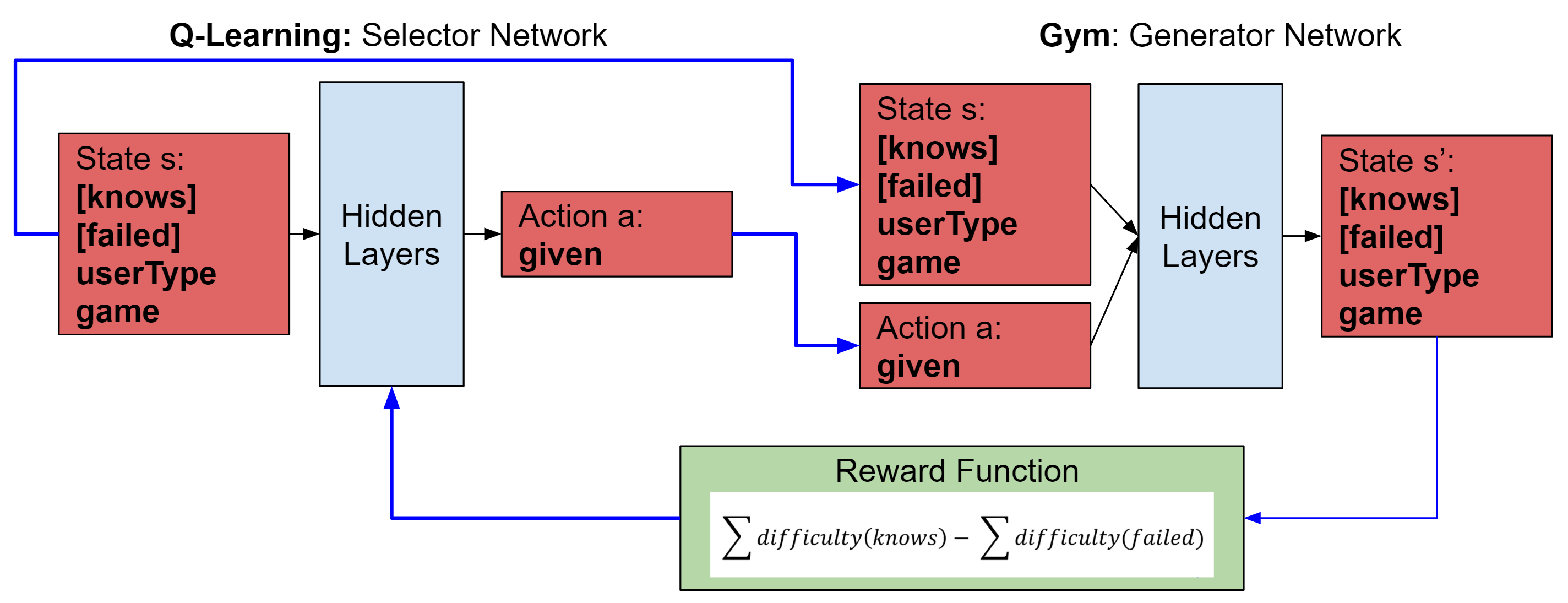


Figure 7 - The system which will decide what words to give to the child upon request

Once this network is trained, it will be queried ten times in a row. Each time, the outputted action will be added to a list. This list will then be return to the Client’s request.

A general note for all of the networks. Currently, it is unknown the true structure of the layers, or how many layers these networks will be. This will require trial and error to get better results.

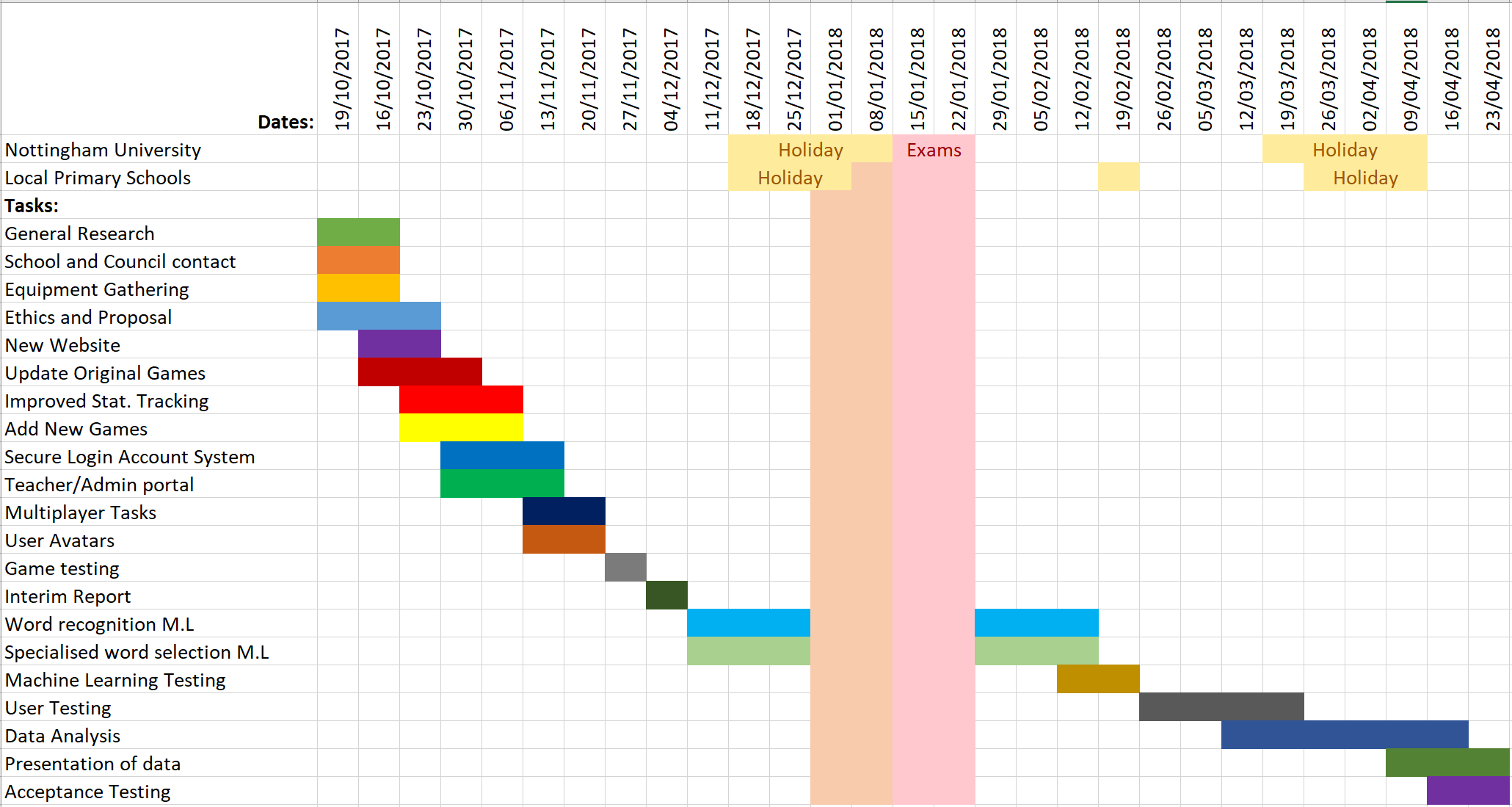
**Other monitored statistics**

On top of the statistics we will collect for the teacher feedback, and neural network training, we will also be collecting data to evaluate the games, and then make modifications appropriately. We will collect information such as average playtime, average scores, and generate a heatmap of where a child clicks on the screen.

We will be able to analyse these information to make changes to the games themselves such as :

* Change the difficulty of the games
* Changing controls
* Changes to images if there is too much ambiguity
* Improve running performance
* General Bug Fixes

# Progress

The initial project work plan was made up of an overall Gantt Chart with specific milestones and deadlines expected to be achieved at specific weeks. Tasks were allocated to specific members with an aim to minimise possible bottlenecks with un-utilised members waiting for others and integration points between members individual work. With only three members there is no single elected group leader to track each person’s progress and spot delayed items early.

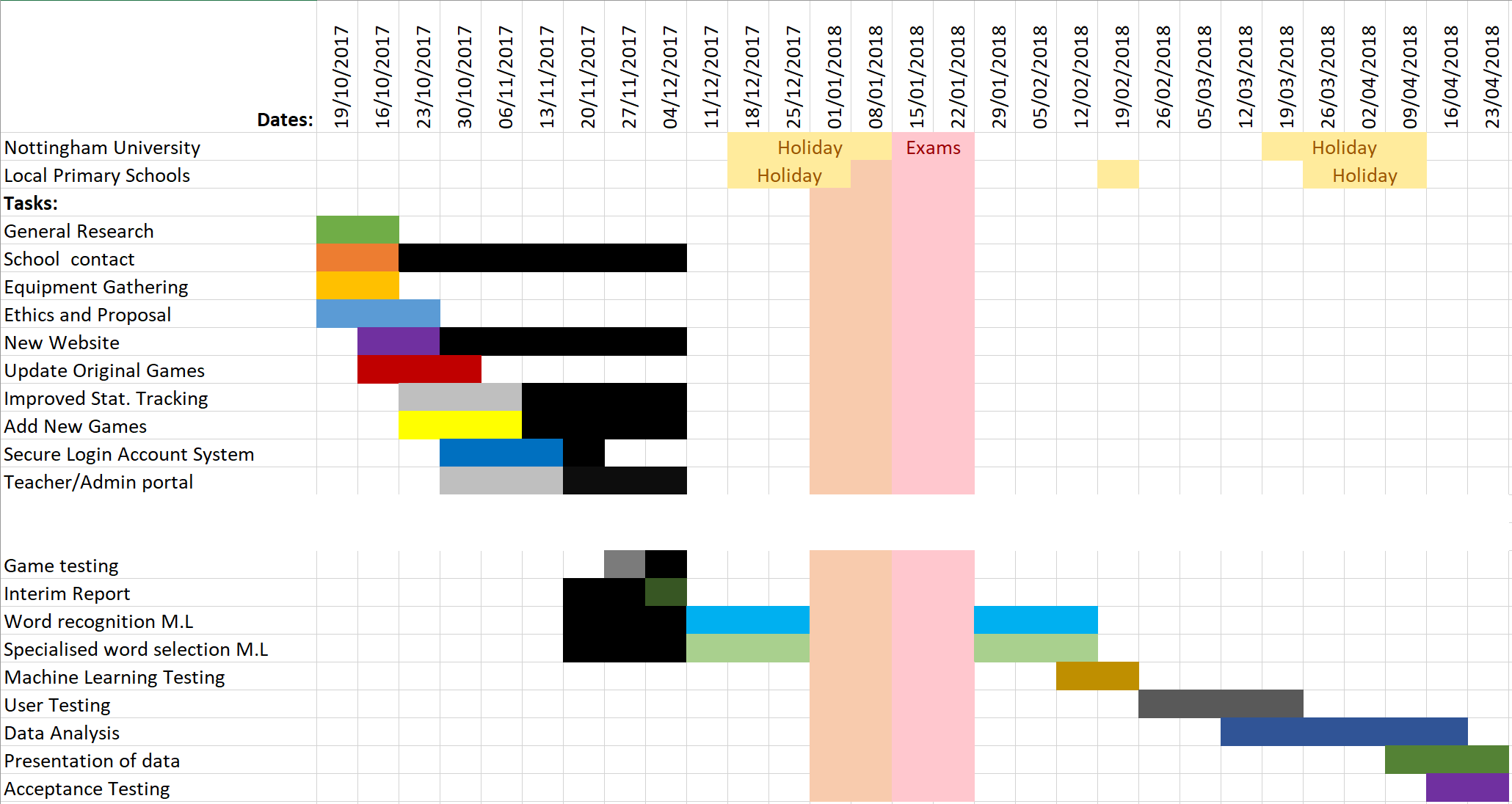


Figure 8 - Original Gaant Chart

Figure 9 - True Gaant Chart. Black Bars indicate the task's time of when it was actually worked on. Multiplayer and User Avatars removed due to time constraints.

Comparing the initial Gannt Chart with the updated Gannt Chart there have been several modifications and delays. Firstly, the time required to complete the new website and server-side was greater than originally thought and this the time required has been extended to accommodate this. The time allocated to update the previous Savannah School games and create a number of new games was flexible. The games represent a vehicle for collected statistical information, as such they only need to meet an adequate standard to achieve this, the faster and easier they were to develop the more would have been made. Development is behind what was originally planned but ahead of the minimum required for Machine Learning purposes.

To compensate for the delays with web development Machine Learning research, planning and implementation has been brought forward, in order to prevent a future bottleneck where rapid significant progress would be required to not delay field testing of the overall system.

Behind the games would sit a Word API and a Statistics API to handle interaction and information transfer between the games and the server. This is to be started in the next few weeks and to be completed by mid-January. As more features are developed the games are continuously updated to make use of this. At present, the words used within each game is hard-coded and the statistics collected but not utilised. Once the API is working the games with be retrofitted to use the given words and send back the statistical information.

An increasingly concerning area is making contact, and organising testing sessions, with local schools; this has proved difficult and time consuming as compared with the original Savannah Schools project, for which a link with a local school pre-existed. The first round of contact, detailed emails regarding the project, generated no response. The second round involves calling schools individually and attempt to locate and speak to a member with appropriate authority. This is currently ongoing.

Tasks are currently more heavily divided with one member leading each area of development, such as web development, games development or Machine Learning. With contributing to each area being monitored by that member, this is with an aim to improve transparency, monitor progress and locate bottlenecks more easily. The original project plan was optimistic with a relaxed view towards goals, deadlines and alternative options; the current plan includes stricter goals and more attention on knock-on effects of delays.

# Contributions

Oliver’s roles included updating the old Savannah Schools games to follow common themes and structures alongside creating a number of new games to increase the variety and learning areas covered. This was completed in Phaser in HTML5 within the Phaser Editor IDE, both of which had to be learnt during the project. He was assigned to game development as he had the most exposure to Human Computer Interaction. Oliver also conducted research into local schools before making the first round of contact. He was responsible for creating an API to handle interaction and information sharing between the from and back end alongside retrofitting all games to include statistical tracking to collect information on each user’s game sessions.

Rob’s contributions included initial project research, built on his previous involvement in the Savannah Schools project. He also created graphical items for the new games, alongside improving the graphics in the previous ones to ensure uniformity and a common standard. Rob lead the research for the Machine Learning elements, resulting in a Machine Learning plan from which the practical implantation will be based. He was assigned to this area as it matches his personal interests and he had the most existing knowledge on the topic.

Lawrie’s responsibilities included developing a website front end, from which the users can play the games and teachers can view statistical information. This was done in HTML and Node.js, with Node.js having to be learnt for the project. None of the web content from Savannah School could be kept and Lawrie needed to create the entire system from scratch. He was assigned to web development as he had considerable experience with front end development and would be able to complete this fairly quickly and to a high standard.

# Conclusion

# Reflections

**Oliver:**

At this stage I have been tasked with updating and improving the existing games, creating a series of new games from scratch and reaching out to local schools in order to trail the project in a real classroom environment. Retrofitting the old games was fairly straightforward once I became accustomed to Phaser Editor, which itself is quite similar to Android Studio which I have used before. I feel my previous experience with HTML did not aid me as much as thought it would have, as the games are created in ordinary JavaScript with no modifications or adaptations for web development.

The HCI aspects of game development continue to be enjoyable, originally brainstorming features and discussing the planned User Experience was also interesting. Behind the games would sit a Word API and a Statistics API to handle interaction and information transfer between the games and the server. I am confident both of these will be completed on-time and possess all the required functionality to get word sets from the server and return feedback based on the users play session.

I carried out the research and first round of contact with schools in the local area in an attempt to locate a suitable candidate. The research was simple and information freely available. However the response from the schools, or lack thereof, was discouraging. I felt the original contact should have led to some interest in trialling the project based on its potential benefits, but that was not the case.

**Rob:**

I initially spent the first half of the semester working on graphics and design work for the project. I felt this was important to do first in order to have clear direction for what we wish to do later. Once this was completed, it became clear that the original plan of waiting until the end of the semester before even exploring the machine learning component was un-wise. During all of the design, and initial thoughts about the project, the machine learning was seen as a black box which would take in information and then give back a choice for ten words. It was not clear how we would do this.

I believe that the identification of this potential bottle neck, along with the ability to reflect on my current progress, and the progress of the project enabled us to adapt to this new change. The Machine Learning components were something I began to initially research by myself, and I began to thoroughly enjoy it. I began to read lots of papers and watch many videos on the topic. My interest in Machine Learning through self-study has lead me to wanting to peruse a future career in the area.

If I were to do this project again, I would ensure that all elements of the system were thought through fully before development began, so a more accurate plan could be created. Assuming things to be black-boxes can help with simplicity, but it will mask their potential problems.

# References

https://research.google.com/pubs/pub45924.html

https://medium.com/@awjuliani/generative-adversarial-networks-explained-with-a-classic-spongebob-squarepants-episode-54deab2fce39