Machine Learning Techniques for Avalanche Prediction

Created by: Liam Kenny
URN: 6441229
Supervisor: Dr. Alireza Tamaddoni-Nezhad
Module: COM3001 Professional Project
Department: Computer Science - Faculty of Engineering and Physical Sciences

TABLE OF CONTENTS

DECLARATION OF ORIGINALITY

"I confirm that the submitted work is my own work and that I have clearly identified and fully acknowledged all material that is entitled to be attributed to others (whether published or unpublished) using the referencing system set out in the programme handbook. I agree that the University may submit my work to means of checking this, such as the plagiarism detection service Turnitin® UK. I confirm that I understand that assessed work that has been shown to have been plagiarised will be penalised."

ABSTRACT

Mountains contain some of the most extreme and unforgiving terrain known to man, yet each year more and more people take winter breaks for skiing, adventure and relaxation. Many mountain towns have exploded in capacity in order to accommodate these tourists yet the controls used to protect the towns and people within them have not changed in many years. Mountain resorts rely heavily on expert knowledge for the prediction and control of avalanches. Although largely successful, there is always room for improvement when it comes to the protection of human life.

A computer system has been developed in order to address this, combining two Artificial Intelligence based computer programs, the NXD2000 and NXD-REG [1]. Both of these programs use nearest neighbour algorithms for the prediction of avalanches across mountain regions. This system is the only one of its kind which has been implemented in mountain resorts.

This project addresses the need for improved avalanche prediction mechanisms and explores the possibility of optimising this through the use of machine learning techniques. There are many potential learning techniques which could be applied to avalanche prediction but the focus of this project will be towards neural networks. These will be evaluated and compared to the nearest neighbour classifiers which are used by the NXD prediction systems.

ACKNOWLEDGEMENTS

I would first like to thank Utah Avalanche Centre for providing me access to their entire data archive free of charge. This project would not have been possible without their contribution. I would also like to thank the Canadian Avalanche Association. Although their data was not used in the final model, they were very forthcoming with the data they had available. They were also encouraging and enthusiastic about my project which gave me strength when I needed it. My supervisor, Dr. Alireza Tamaddoni was a great help throughout this project. His advice and guidance gave me good structure and kept me on track throughout the project. I would finally like to thank Lauren Page for her constant support and encouragement.

1.0 INTRODUCTION

This report aims to analyse a number of machine learning techniques for the application of predicting avalanches. These predictions will be based on meteorological data supplied by a number of weather stations situated within a single mountain range. The methods used will be evaluated and compared against the existing NXD2000 and NXD-REG systems which lead the way in machine learning based avalanche prediction. Conclusions will be drawn as to how successful each of the learning methods are at predicting avalanches as well as what could be done to improve upon them in the future.

1.1 BACKGROUND

The current standard in avalanche prediction across the world consists of a number of field tests which are carried out by local experts each morning [2]. These are used in conjunction with detailed weather reporting in order to provide an overall avalanche risk rating of increasing severity between 1 and 5. These risk warnings are then issued for a region of varying size, depending on the resort for which the warning is being issued. The tests which are carried out look for weaknesses within the snow pack which may result in avalanches being triggered [2]. These weaknesses are created by snow falling and settling under varying conditions [3]. With precise weather data, it should be possible to understand the snow pack which has developed over a winter season and predict weak layers in the snow and at what point they are likely to trigger an avalanche. Computer systems are capable of modelling scenarios to a high degree of accuracy given a good input dataset. Machine learning algorithms would be able to learn from previous examples of avalanches and find patterns in the weather which contributed to them. This could provide a much higher degree of accuracy when testing for avalanches than the currently used field tests as well as reducing the need for testers to put themselves and coworkers at risk.

A subsection of the Swiss Federal Institute for Forest, Snow and Landscape Research WSL, named SLF has developed two machine learning based tools for avalanche prediction. These are named NXD2000 and NXD-REG and are used together in order to predict high and low risk zones [1]. Both of these systems use weather data fed into nearest neighbour algorithms in order to make their predictions. The NXD2000 makes predictions for specific mountains and localised regions, whereas the NXD-REG makes predictions for wide areas such an entire countries or states. In areas where this has been rolled out, the NXD systems act as a supplementary resource to the field researchers rather than a replacement. These systems have been implemented in a number of high profile ski resorts around the world but unfortunately it has been too costly to roll these out to the majority of locations. This means that most resorts are left with only the traditional and highly dangerous method of predicting avalanches in the field. To this day, avalanche forecasters provide a higher success rate of avalanche prediction than computer systems alone [4].

1.2 PROJECT BENEFITS

This project aims to explore and evaluate different methods of machine learning applied to avalanche prediction. A better insight into prediction methods will help to improve the progression of computational based avalanche prediction. Not only will such advancements help to reduce the risk to life that avalanche forecasters take on a daily basis, but it will also provide improved safety for all mountain users. Computational modelling and prediction systems will undoubtably be the future of avalanche prediction and it is hoped that this project will assist on this journey.

1.3 AIMS AND OBJECTIVES

The overall aims of this project are to:

- Research and document the causes and effects of avalanches and the prediction methods that surround them.
- Evaluate the benefits and problems with current avalanche prediction systems.
- Research and compare neural network algorithms for use with weather based avalanche prediction.
- Acquire weather and avalanche data for a localised area.
- Design and build a neural network based system which will accept weather data, extract the relevant features from the data, and output a prediction of whether an avalanche will or will not occur on a given day.
- Evaluate the system and compare its success rate to those of the current prediction methods.
- Assess the suitability of the learning mechanisms covered in this project for the application of avalanche prediction.

1.4 SUCCESS CRITERIA

In order to assess the success of this project, success criteria will be used during the Project Evaluation. For this project to be successful, each of the following requirements must be met:

- Research must have been conducted in the field of neural network based architectures for the application of weather based predictions.
- A working neural network which makes avalanche predictions at a comparable validation accuracy to previously published results.
- The accuracy and operability of the application must be evaluated and compared against the current NXD prediction systems.

1.5 POSSIBLE PROBLEMS

The proposed machine learning based system will be designed around a neural network architecture. Neural networks require training, testing and live use. This means that three separate datasets will be required which contain no overlap. Acquiring weather and avalanche data of this volume may be problematic as such data is not often readily available to the public. Furthermore, the success of the neural network will rely heavily on the quality of the data which is fed into the system. Avalanches by their nature are in remote areas which are not often patrolled by mountain guides or avalanche forecasters. Therefore there will be some loss within the avalanche data provided, as well as inaccuracies with the timings and details of avalanches. This will undoubtedly reduce the accuracy of the system, however perfect avalanche data is not possible to acquire for professionals and this is a problem which will be faced in the real life application of such a system. Finally, the data fed into neural network will be processed from its raw format in order to extract the most relevant features. In order to achieve a successful network, these features need to be relevant and optimal. Some work will be required in order to determine the best features to feed the system and the quality of such features will impact the accuracy of the whole system.

1.6 PROJECT MOTIVATION

The project which I have chosen combines some of my closest personal interests with the most fascinating area of my course. I learned to snowboard when I was 10 years old and have been infatuated with it ever since. I learned with my father and our trip to the mountains would easily be the highlight of each year. As we progressed, we took an interest in 'the backcountry', with the freshest, deepest snow, away from the busy slopes which most associate with the sport. With this excitement came risks; un-patrolled runs meant a higher chance of avalanche and crevasse. In 2015 me and my father both took our Avalanche Safety Training (AST) Level 1 qualification, followed by numerous trips to the backcountry - each of which involving further avalanche training. In 2017, while snowboarding with a local mountain guide, my father was caught in an avalanche which had a vertical range of 150 metres and span a width of 50 metres. With the help of his training as well as incredibly quick thinking, he narrowly avoided losing his life. We have never taken the mountain for granted, but this unsettling experience left both of us shaken and I am determined to do what I can to improve avalanche safety for all mountain users.

Artificial Intelligence and Machine Learning have been the most interesting topics which I have explored during the course. My love for programming was put to the test with an AI module which I took in year 2 of my degree, but my motivation was sparked more than ever. The multitude of applications and seemingly limitless capabilities for AI gave me an interest in Computer Science which I had not found before then. I can see machine learning improving avalanche safety far beyond where it currently lies and becoming the future standard of prediction. I want to be at the forefront of this progression and explore new ways that computing can help to improve the success of avalanche safety.

2.0 LITERATURE REVIEW
2.1 SNOW AND AVALANCHE SCIENCE

Snow crystals are formed in temperatures below 2°C when tiny particles of dirt accumulate moisture, ever increasing in size as they collide with each moisture particle until they reach a terminal weight where they are heavy enough to fall to the ground in the form of a snowflake [5]. The properties of snow are defined mostly by the conditions present during their formation, the largest contributors being air temperature, wind speed and rate of precipitation [6]. As the snow settles, it forms layers of different snow with different characteristics from each snowfall. These layers build up to form what is known as 'the snowpack'. In a similar way that the rings of a tree provide historical growth information, each layer provides insight into the history of this season's snowfall [7].

For an avalanche to occur, four parameters need to be present: a slab which creates the mass behind the avalanche, a weak layer for the slab to slide across, a slope for the snow to fall and finally a trigger [8]. In practise, snowpacks consist of many layers of different properties and it is difficult for experts to fully understand the nature of the snowpack because of this. As seen in Figure 1, snowpacks which are said to be stable contain their most dense snow on the base, providing stability and grip [9]. Fresh snow will sit above this and with its light weight, any unstable snow will fall straight off the slope when humans or animals pass through it. Unstable snowpacks have the opposite density distribution to that of a stable snowpack, with heavy, hardened snow on the surface and a weak layer below. The heavy top layer will, with the right slope and trigger, slide across the weak layer resulting in an avalanche.

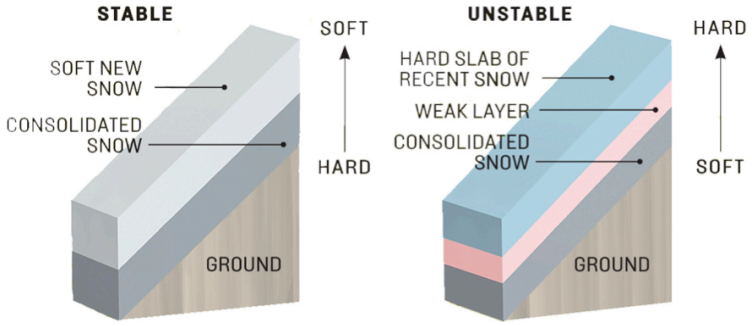

Figure 1 - Layers of a Snowpack - [9]

This simplified explanation defines the core components of an avalanche, however the National Avalanche Centre and American Avalanche Association state that there are nine separate types of avalanche with different properties [10]. These fall into two main categories of slab and loose avalanche, where snow falls down a slope in solid blocks or broken flurries respectively. While these all result in very different outcomes and are products of different conditions, each still requires the premise of a heavy layer sitting on top of a weaker layer to trigger.

Due to the complexity of analysing a snowpack across varying slopes and with the potential for numerous types of avalanche, avalanche prediction is a very difficult task. Not only does it require requiring many different variables to be considered at once but also an expert understanding of how each contributor will effect the overall risk.

2.2 CURRENT PREDICTION METHODS

Avalanche prediction is a crucial task in the mountains and is relied on heavily for winter sports, construction planning, road closures and traffic flow as well as insurance provision for permanent structures [4]. Understanding and predicting avalanches allows for mountain planners to safely conduct controlled avalanches before they cause harm to people and infrastructure. Avalanche paths are first evacuated and then slopes are artificially triggered with the use of explosives [11]. Therefore the accuracy of such a prediction needs to be highly precise and err on the side of caution. Without accurate predictions, all mountain users could be at risk.

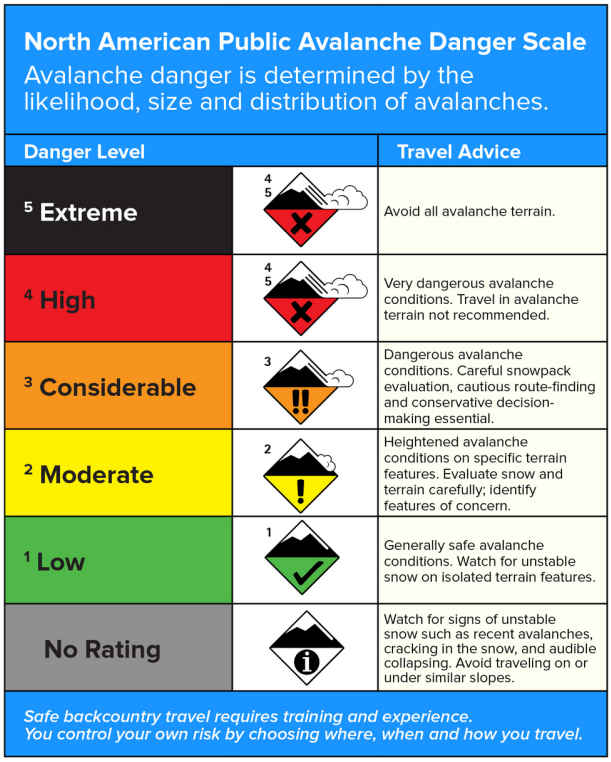

Figure 2 - Avalanche Danger Scale - [12]

Traditionally, avalanche prediction is a task carried out by field experts who venture into dangerous avalanche terrain in order to ascertain the likelihood of an avalanche occurring, as well as the probable scale of such an avalanche. These two factors combined provide experts with a way of defining a danger level for certain areas of terrain [12]. The data used to make these predictions is an amalgamation of the weather conditions which have been recently present as well as results from the field tests carried out each morning [11]. An avalanche warning is then given for a localised area, usually a single ski area consisting of a small number of mountains. A danger rating between 1 and 5 is then provided for the given area. An example of such a scale can be seen in Figure 2 [12].

Field tests are not only time consuming and require extremely specific expert knowledge and skill, they are also highly dangerous, requiring testers to venture into the avalanche danger zones to carry out their tests. Tests will include spotting avalanches which have occurred since the last time testing was done, providing information on how the snow has been behaving recently. Additionally, snowpack analysis will inform testers the structure of the snow at each level of the snowpack. Finally, extended column tests will allow testers to see not only where weak layers in the snowpack are but also how much force is required to trigger an avalanche and how large that avalanche is likely to be [13]. The data collected from field tests provides a more detailed view of the snowpack than could easily be extracted from weather data alone.

Weather data for the present day also needs to be incorporated into the findings of the testers before their final warning level can be given. Weather data for the current day is important as it

provides an idea of what is likely to happen to the snow which has just been observed in the field [14]. For example, sunshine will bring warmer temperatures for southern facing slopes, causing thawing of the upper most layers. Alternatively, an Easterly wind will weaken an east facing slope while loading a west facing slope with large amounts of fresh snow [15]. These factors are likely to impact how an avalanche is triggered and therefore what areas mountain users should avoid. Once all of the above data has been interpreted, the overall risk level can be calculated. Each avalanche centre will calculate their risk level and publish this for their mountain region.
Computers can handle large amounts of complex data very quickly and machine learning techniques provide a mechanism for processing such data and providing predictions based on previous examples. For this reason, the use of machine learning for avalanche prediction is a logical progression in the field of snow safety and one which this project aims to explore.

2.3 NXD SYSTEMS

2.3.1 NXD ARCHITECTURE

Although field tests are relied on heavily for almost all avalanche prediction, the information gathered as a result is simply a clear view of the weather history, in addition to a real-life avalanche simulation under varying conditions [13]. They are very useful for manual predictions as it is incredibly difficult, if not impossible, for humans to extract an accurate understanding of the snowpack from weather data alone. There are simply too many variables forming complex relationships in order to accurately predict the snowpack. However, as computing power has increased and weather data has improved, it is becoming possible to predict avalanches through the use of machine learning techniques.

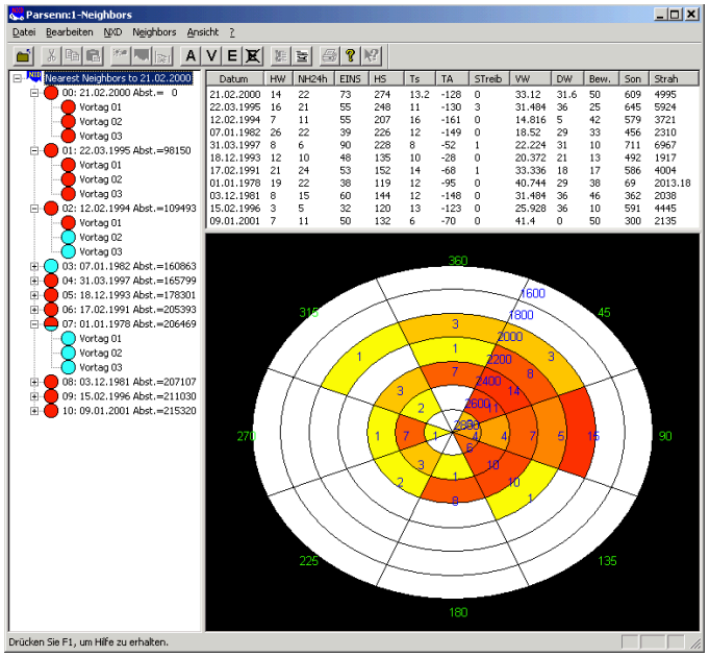

Figure 3 - Result of NXD-2000 showing nearest neighbours, input data and diagram of observed avalanches [1]

The first published development of avalanche predictive machine learning was in 1983, where Buser developed the first NXD system [1]. NXD used a nearest neighbour model in order to find the most similar days based upon meteorological data to the current day. Figure 3 shows an example output which is displayed to the user of the NXD-2000. This would allow the user to not only see the days which were similar but additionally whether or not avalanches had occurred on those days and detailed information about those avalanches. In 1992 the Swiss Institute for Snow and Avalanche

Research (SLF) joined with Buser and a number of other academics to work on improving the original NXD system. The system which they developed was known as the NXD-2000 and focused on improving the flexibility of the original system, allowing it to be applied to different locations and incident scenarios such as debris flow and forest fires. Additionally, the system was adapted for Windows95, allowing users to retrieve information from the NXD-2000 system and use this within other useful applications such as Microsoft Word or Excel [16].

When publishing avalanche warnings, the destructive level must be considered; taking into account the magnitude of a potential avalanche as well as what assets may be impacted as a result. In order to make accurate predictions it is important for forecasters to know not only what conditions were present when avalanches occurred in neighbouring days, but also information describing the nature of such an avalanche. For this reason, NXD-2000 provides information on the size, location, type and damage caused by each avalanche as well as any control measures which were implemented [16]. It is also possible for users to query the system for detailed information regarding weather or avalanche data [1]. At the time of its development, this was by far the most informative tool for avalanche forecasters to use. More detailed information means that forecasters can make a more informed decision about the hazard level to be issued and what controls are required in order to ensure safety.

The NXD-2000 prediction system uses meteorological data from a number of weather stations in a localised area. Due to the nature of snow formation, some variables have more of an influence on the snowpack, and therefore avalanche risk, than others. Work by McClung and Tweedy (1993) helped to understand the correlation between individual variables and the occurrence of avalanches while Boyne and Williams researched the influence of meteorological variables on avalanche formation [16]. This work helped to form the selection of variables to be considered when finding nearest neighbours. Obled and Good first detailed the use of elaborate variables which attempt to derive new information from raw data [1]. Kristensen discovered that variables such as temperature, windspeed and snow height have critical transitions, meaning that variation at some ranges produces a larger effect to the avalanche risk than others [1]. For example, a temperature change from 0°C to +5°C makes a much greater difference to the snow properties than a change from -25°C to -20°C [16]. NXD-2000 uses a hyperbolic function to achieve the desired transformation of temperatures. This is defined using the following function where TS is the snow temperature.

Eq. 1	$TS' = 20*\tanh(0.2 * TS)$	[1]

Another variable used within NXD-2000 was Settlement (*St*), which describes the way in which the snow has changed over time since it fell and is calculated using the following formula, where *HS* is the height of the snow and *NS* is the amount of new snow which has fallen.

Eq. 2	$St = HS_0 - HS_{-1} + NS_0$	[1]

In the case of the NXD-2000, many of the elaborate variables used require data from the three previous days, known as ‘pre-days’. The reliance of these variables creates a problem when data is missing from the system. Should a single day of data not be available due to an outage, bad connection or any other factor, there will be no prediction available for the next three days. Instead, the system must wait to acquire three complete days of data in order to continue making possible neighbours.

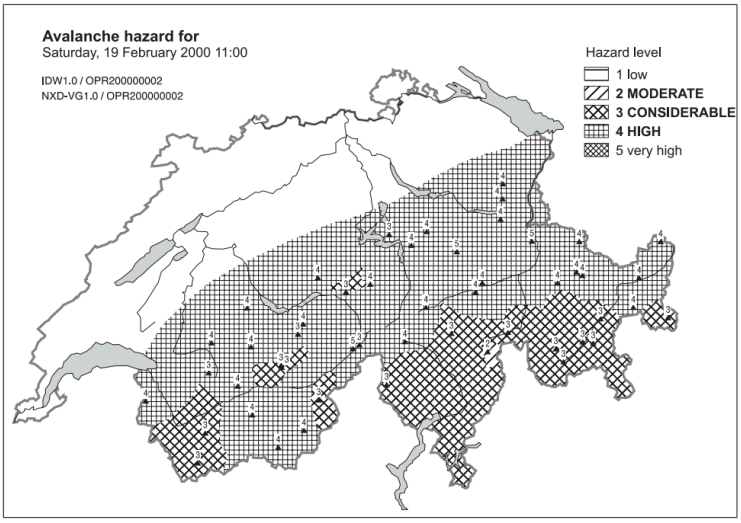

Figure 4 - Result of NXD-REG for 19/02/2000 [1]

Furthermore, the meteorological data provided by weather stations is compressed by taking an average of the day’s readings. Large changes in temperature, wind speed and other variables can occur very quickly in the mountains and cause drastic changes to the snowpack [17]. The result of averaging across a 24 hour period means that information is lost and never considered by the system, causing valuable and readily available data to be overlooked. However, the accuracy of detailing avalanche occurrences cannot be narrowed to below a 24 hour period. This limiting factor means that weather data cannot be correlated to avalanche occurrences at a higher degree [17].
In addition to the development of the NXD-2000, another system was developed by SLF, Bruser and their team named the NXD-REG. This system, instead of predicting avalanches, provides a suggested avalanche risk rating based on a combination of the risk levels for each of the returned neighbours. Each of the 59 weather stations of the Swiss observation network are queried, examining their complete 10+ year dataset [1]. NXD-REG is designed for regional forecasting and its predictions cover areas such as counties, states or even countries [18]. The data provided by NXD-REG is a suggestion to avalanche forecasters of a probably forecast for their region. An example output from the NXD-REG system can be seen in Figure 4.

This system alone is not sufficient for forecasting avalanches or providing an informed risk level for localised regions. NXD-2000 provides a much more detailed insight into the risks than NXD-REG. However, in practise NXD-2000 and NXD-REG are used in conjunction with traditional methods of avalanche forecasting in order to create the comprehensive report which is delivered at 8am each morning [18]. These prediction systems act as a supplementary tool for avalanche forecasters rather than a replacement.

2.3.2 NXD DISADVANTAGES

As discussed above, a nearest neighbour learning model has been applied to the NXD-2000 and NXD-REG systems, built by SLF in the late 1990s. Since these were developed, they have been at the forefront of machine learning avalanche prediction systems and have shown the avalanche forecasting industry the power of machining with respect to avalanche prediction. However, there are a number of problems with these systems which are limiting their success.

- The elaborate variables used within the system are calculated using pre-days; the meteorological data from the n previous days. The largest number of reliant pre-days is 3, meaning that if a single day of meteorological data is lost, the model cannot make predictions from the day of the missing data or the three following days [16].

- Variables are averaged across a 24 hour period and predictions are made on a daily basis. The accuracy of variables cannot be reduced due to the relationship between the meteorological data and the avalanches themselves. As it is not possible to tell when an avalanche occurred to less than a 24 hour period. Any changes in data at a shorter period than this therefore cannot be attributed to a particular avalanche. Although this is the case, snow conditions often encounter extreme changes in rapid succession. Such changes can contribute to vastly different formations of snowpack and therefore greatly effect the likelihood of an avalanche [16].

- Avalanche control on previous days is not considered when calculating risk ratings. Intentionally triggering avalanches is the most effective method of reducing avalanche risk. Potentially dangerous avalanche paths are triggered in a controlled way usually using explosives. On slopes which have been triggered, there is a negligibly low chance that an avalanche would occur. Not considering avalanche control within calculations means that important information about the snow is not utilised, limiting the success of the system [16].

- The definition of an avalanche day is unclear. All days which are considered as ‘avalanche days’ by the system are treated in the same way. A positive value for an avalanche day will simply suggest that an avalanche is likely on a similar day. However, this system does not take into consideration the magnitude of the avalanche which occurred. For example, a 10m x10m avalanche may not be recorded as an avalanche within the database as the observer may have considered this to be too minor. A 20m x 20m avalanche may be considered and noted as an avalanche day, in the same way as a 100m x 100m avalanche. It is clear here that the risk to life in each of these cases is vastly different, yet there is no definition of way to distinguish between different sized avalanches [16].

- The weighting of meteorological variables is difficult to correctly calculate. The NXD systems discussed above require the input variables to be weighted according to their effective contribution to an avalanche. These weightings have been calculated through trial and error,

arriving at a set of weightings which seem to correlate best to the training data of the system. However, the accuracy of these weightings may be incorrect and with a range of non-linear variables, it is very difficult to ensure that these are as correct as possible [16].

Each of these problems contributes to limiting the success of the NXD systems. Since the NXD systems were first developed, an array of new machine learning techniques have surfaced. The use of different machine learning techniques will be discussed in the next section, outlining the potential advantages and drawbacks of each with respect to avalanche prediction. Furthermore, dataset structure, cleaning and normalisation will attempt to provide methods to reduce the impact of these problems to avalanche prediction systems in general.

2.4 LEARNING ALGORITHMS

2.4.1 ARTIFICIAL NEURAL NETWORK (ANN) ARCHITECTURE

Artificial Neural Networks (ANN) have become increasingly popular in recent years due to their similarity to the neural structure of an organic brain. They are able to make decisions and draw conclusions based on complicated datasets even when these datasets contain noise, are incomplete or contain irrelevant information [19]. As the name suggests, the structure of an ANN is comprised of many interlinked artificial neurons, named perceptrons. Each of these perceptrons receives a range of inputs, processes these inputs based on its individual configuration parameters and produces a resultant output [20]. One of the more popular configurations of neural networks is to arrange perceptrons in layers where each node of the previous layer feeds into each node of the next. This is known as a Multi-Layer Perceptron (MLP) network.

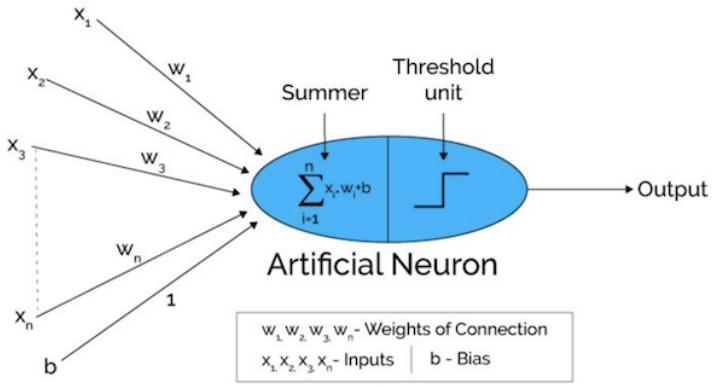

Figure 5 - Structure of a perceptron - [22]

The structure of a perceptron is to take a number of inputs, each of those with a variable weight applied to their value. The sum of these inputs is calculated along with an additional variable named a bias. If the result of this calculation is greater than a given threshold then a 1 will be output, otherwise a 0 is returned [21]. Figure 5 clearly displays the structure of a perceptron with each of its inputs and outputs. Within an MLP network, an input layer will accept a number of input variables and process these. Perceptrons within this layer will feed into the subsequent hidden layer(s). These layers can contain a wide range of nodes and this is defined during the setup of the network. The final layer of the neural network is known as the output layer. This layer will contain the number of different categories which the system is attempting to classify [22].

During the setup of an ANN, the weight and bias variables for each node and edge within the network are set randomly. These are then adjusted during training using a method named Back Propagation. When using a supervised learning method, the training is done using known, labelled data. With this information, the success of the predictions provided by the output layer can be evaluated. Evaluation is carried out using a cost function which measures how good the prediction was [22]. With this error value, Back Propagation is performed, changing the weights within the network in an attempt to reduce the error value. With multiple error rates, it is possible to calculate the gradient of the slope which connects these two errors. As the goal is to minimise the error of the program, the weights are shifted in the direction of the negative gradient by a step size defined by the Learning Rate (LR). This method of optimisation is known as Gradient Decent [23].
After varying the weights of the network, eventually a gradient of 0 will be reached for an error rate in the program. This point is known as the local minimum of the function, meaning this is the most optimal solution for parameters in a local range [24]. However, the local minimum may not always be the same as the global minimum of the function and the most optimal solution.

In the following section, different approaches for finding the global minimum of parameter weights will be discussed. This automated method of finding optimal weights for system parameters is likely to be more accurate than the trial and error approach which is used in the NXD-2000's KNN model.

2.4.2 GRADIENT DESCENT METHODS

Gradient descent is the process of gradually changing the weights of different network parameters in order to optimise the accuracy of a machine learning (ML) model. Batches of training data are used and the average accuracy of the model is taken. This accuracy is input to an error function which is then used during back-propagation in order to adjust the model's weights. The magnitude of the changes made at each stage is known as the learning rate. Generally, smaller learning rates are considered to perform better than larger learning rates as they gradually but consistently gravitate to a local minimum however there are advantages and disadvantages to each.

Smaller learning rates may take longer to find a minimum than larger learning rates as they make only small changes to the weighting of the model parameters. However, by taking smaller steps, they are less likely to bounce between too high and too low a weight for a given minimum. Instead, they tend to move slowly but consistently towards their local minimum. This is displayed above in Figure 6. However, by having too small a learning rate, the gradient descent algorithm will become

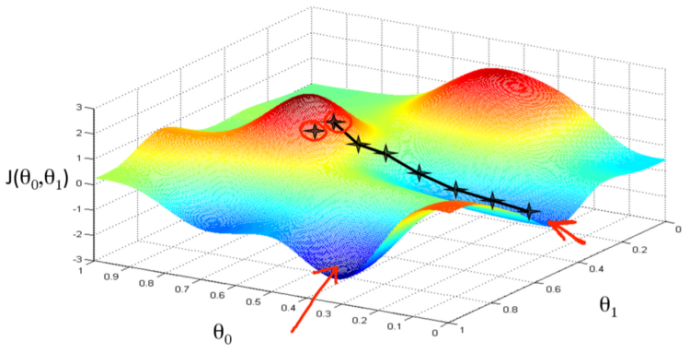

Figure 6 - Visualisation of gradient descent- [24]

stuck in a local minimum more easily. Larger learning rates can produce a more chaotic method of learning, over and undershooting the local minimum. If the learning rate is too large, it may never find the local minimum at all. However, larger learning rates are much less likely to become stuck in local minimums, meaning that they are more likely to find a global minimum.

A number of methods of gradient descent methods will be tested and compared during the building

Eq.3 $$ht = sigm(Whxxt + Whhht−1)$$ [25]

of the prediction system. As each method will be more suited to different types and structures of data, it is important to test these with the actual data used within the system.

2.4.3 RECURRENT NEURAL NETWORKS (RNN)

Recurrent Neural Networks are a type of feed-forward neural network which contain loops in order to cary information forward from one iteration of the network to the next in the form of activations [25]. These types of network prove useful for datasets which are sequentially related. The general equation for which is used to calculate node outputs is as follows:
Within a standard RNN, data from the previous prediction is passed on, as seen in Figure 7. The way amount of data which is stored and passed on can be configured manually. When back-propagation occurs in this type of network, the error within the carried forward data weights tend to exponentially increase [26]. Not only can this lead to training taking a very long time but it can cause inaccuracies in predictions when data dependencies span a long period of time [27].

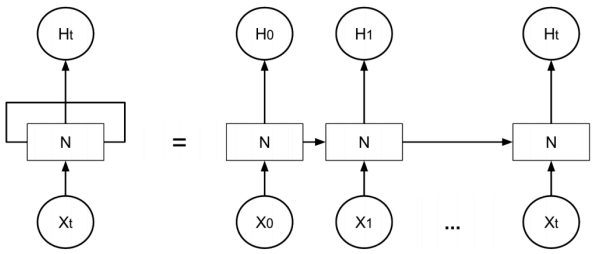

Figure 7 - Structure of RNN - [25]

However, RNNs have proven to be successful for a number weather prediction systems due to the sequential nature of weather [25]. In order to avoid the problems mentioned above, these systems make use of a specific RNN architecture used to extending the persistence of historical data throughout multiple runs. This was first introduced by Hochreiter and Schmidhuber, named Long-Short Term Memory (LSTM) [26].

2.4.4 LONG SHORT TERM MEMORY (LSTM)

LSTM is an RNN architecture which attempts to minimise the problems associated with error back-flow through the use of an efficient, gradient-based learning algorithm [26]. LSTM makes use of memory cells and gate units in order to pass data from previous iterations to the current iteration of the algorithm. This method provides an easier way to utilise data from very large sequences in each iteration. Data in excess of 1000 iterations can be used with LSTM [26].

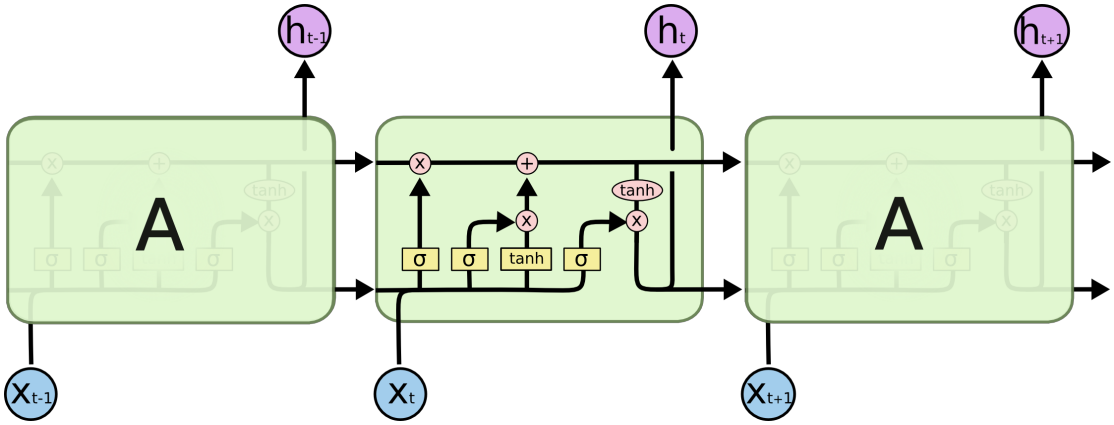

Figure 8 - Structure of LSTM - [26]

The reason that an LSTM network is able to do this is because, unlike a standard RNN, it uses memory blocks called cells to store information. These cell are modified through the use of adders or multipliers during each iteration. As these cells are not changed completely, but instead modified slightly through each iteration, the program is able to decide what information to save and pass on, and what information to forget [28]. The cell state acts as a conveyer which keeps the relevant information separate from the flow of predictions [29]. This flexibility gives the LSTM architecture an edge over a standard RNN architecture.

As seen in Figure 8, there are three separate gates which are used to extract and store information in the LSTM cell state: Input Gate, Forget Gate and Output Gate. Each of these cells has a similar structure, containing a sigmoid function (σ) which decides how much information should be processed and a multiplicative operator which concatenates the new information with the old. The gates described below accept two values each: $\boldsymbol{h_{t-1}}$ and $\boldsymbol{x_t}$

- $\boldsymbol{h_{t-1}}$: the prediction made during the previous iteration
- $\boldsymbol{x_t}$: the input of the current iteration

Forget Gate

The first gate which is reached is the Forget Gate. This is used to decide which values are no longer of use and should be removed from the cell state [27]. The output from the previous iteration and the current input are concatenated and fed through a sigmoid function. An output of 1 from this function states that all of the information should be kept and a value of 0 states that none of the information should be kept [26]. The equation for this function is as follows:

Eq.4 $$f_t = \sigma (W_f \bullet [h_{t-1}, x_t] + b_f)$$ [26]

As can be seen in this equation, the gates contain an individual weight and bias which are adjusted through the training of the network. Finally, the output from this gate is multiplied to the cell state, removing information which is no longer relevant.

Input Gate

The next gate within the LSTM structure is the Input Gate. This regulates what information is saved to the cell and therefore passed on to further iterations [27]. This gate contains two separate parts, the Input Gate Layer and the tanh( ) layer. Similarly to the Forget Layer, the Input Gate Layer consists of a sigmoid function. A value of 1 indicates that all of the information is passed on and a 0 indicates that none of the information is passed on [26]. The equation for this step is as follows:

Eq.5 $$i_t = \sigma (W_i \bullet [h_{t-1}, x_t] + b_i)$$ [26]

The tanh( ) function crates a vector of possible candidate values which may be passed onto the cell state. The equation for this step is denoted as:

Eq.6 $$C'_t = tanh(W_C \bullet [h_{t-1}, x_t] + b_C)$$ [26]

The outputs of these two functions are combined using a multiplicative function [26]. This combined output is directly added to the cell state and this new information is passed on to the next iterations of the program [27]. The equation for this addition to the cell state is defined as:

Eq.7 $$C_t = (f_t * C_{t-1}) + (i_t * C'_t)$$ [26]

Output Gate

The last gate within the LSTM structure is the Output Gate. This produces an output for the current iteration of the algorithm. The Output Gate also contains two steps, the first of which is a sigmoid function, applied to the output from the previous iteration and the current input. This step calculates which parts of the cell state will be input and which will not [26]. The equation for this step is as follows:

Eq.8 $$o_t = \sigma (W_o [h_{t-1}, x_t] + b_o)$$ [26]

This output is multiplied with the output from a tanh( ) function applied to the cell state in order to select only the relevant parts of the cell state [26]. The output function is defined as:

Eq.9	$h_t = o_t * tanh(C_t)$	[26]

This output prediction will then be fed into the next iteration of the algorithm, at which point this process starts again.

The ability for LSTM to retain information for a long period of time and use this for predictions makes this method highly applicable to weather based forecasting. The contributing factors for avalanches can span the lifetime of the snowpack and the ability to consider these factors within a neural network makes an LSTM based RNN a suitable architecture for avalanche prediction. The rest of this project will explore the implementation of such a system and evaluate the results against other prediction methods with currently exist.

2.4.5 LSTM ADVANTAGES OVER NXD SYSTEMS

Due to the nature of avalanche and weather data collection, some of the disadvantages which are present within the NXD systems will also be present in an RNN based system. For example, it is not possible for an avalanche to be specified to a smaller range than a 24 hour period. However, many of the disadvantages which are associated with the KNN based prediction systems will be addressed through an RNN.

- The weighting of variables will be calculated through back-propagation, instead of the brute force method of the NXD system, utilising expert knowledge and hand tweaking. Back-propagation provides a method for variables to be automatically optimised through many training steps, aiming to provide a higher prediction accuracy.

- The LSTM architecture provides a mechanism for data across a large time period to be considered within its predictions. Instead of a fixed value of three ‘predays’ being used, as in the NXD systems, varying amounts of data can be held within the cell state for use with predictions. This provides a much more similar method of prediction to that carried out by field experts.

- An LSTM architecture allows the Forget Gate to determine what information is no longer useful for predictions while the Input Gate determines what information should be used in further predictions. Therefore, parameters of whether an avalanche has occurred previously can be utilised in predictions. This includes both artificially and naturally triggered avalanches.

2.5 SUMMARY

As described by S. Bakkehøi in “Snow Avalanche Prediction using a probabilistic method” [4], a range of avalanche forecasting methods have developed, taking into account snow and weather parameters which will influence the release of an avalanche. The most important of these are precipitation, wind and temperature. These will provide an indication of when avalanches are likely occur [4]. The most accurate and effective method of avalanche prediction available today is still carried out by field experts with the support of advanced weather forecasts, readings and daily observations.

Although largely successful, this method is very labour intensive and can be life threatening for those conducting field observations in avalanche terrain. In order to assist experts, two NXD systems were developed using a K-Nearest-Neighbour machine learning architecture. The NXD-2000 provides users with the most similar days in history to that of the current day, based on weather data. Users are able to see not only the days but also what the avalanche rating was on that day, and whether an avalanche occurred or not. These systems make their neighbour decisions largely from the three most important meteorological variables detailed above: precipitation, temperature and wind. However, many other variables are also used to supplement the calculations made by the systems. Although these systems have proven to be highly accurate and useful for forecasters, their structure leads to a number of flaws which limit the success of the system. These include: variables being weighted through trial and error, a fixed range of three previous days to be considered in each calculation and the inability to use whether avalanches were triggered on previous days in the neighbour calculations.

Artificial Neural Networks have become increasingly popular in recent years due to their ability to train themselves, adjusting weights automatically based on accuracy optimisation. This automation means that weights for specific variables and features can be optimised much faster and to a much higher degree of accuracy than making adjustments by hand. Recurrent Neural Networks provide the ability to consider information spanning multiple data items. It is possible in such architectures to feed forward information from one iteration to the next. An LSTM based RNN can extend this, providing the ability for data to be considered within predictions spanning a large range of iterations. Furthermore, this structure allows data to be dynamically stored or forgotten within a cell state, utilised with each prediction. This functionality makes it possible for avalanches being triggered on previous days to be considered as a parameter when making predictions.

3.0 DATASET

The prediction system which will be built alongside this project aims to find trends and make predictions based on weather data, including but not limited to precipitation, temperature and wind. The advantages of LSTM based RNNs over KNN based predictors detailed above highlight the suitability for such an architecture for avalanche prediction.The following sections will document the process of creating and optimising such a prediction system. Each iteration of the program aims to improve the accuracy of the predictions being made. The parameters which contribute to the success or failure of each iteration will be evaluated at each stage in order to provide a clear understanding of why each iteration achieved its respective success.

3.1 STRUCTURE AND CONTENTS

In order for a machine learning system to be successful, good quality data must be used. In addition to this, without a substantial number of positive examples, the system will find it difficult to correctly identify trends within the data. The SLF, creators of the NXD systems, state that it is possible to successfully find trends in weather datasets of three years in size and above [30].

The National Centres for Environmental Information (NCEI) hold historical weather data for 26 countries from many weather stations. This weather data is available to a high degree of precision with a range of over 30 years. The Utah Avalanche Centre (UAC) have collected a vast amount of avalanche data from the entire county of Utah, ranging from 1914 to present. They have kindly made their dataset available for use within this project and this will form the Y label data. Avalanche data has been localised by UAC into 5 separate areas: Logan, Moab, Ogden, Provo and Salt Lake City. The NCEI dataset contains weather stations located in these exact regions. The data collected by these weather stations will be used as the X dataset, and will be mapped directly to the avalanche datasets of each of the corresponding locations.

The weather data from the NCEI contains a large number of readings based on temperature, precipitation, wind and more. An example of the variables available are shown here in Figure 9. Each of these variables are recorded hourly, 365 days a year. Some of this data will be more useful than others and the dataset used within each implementation method may vary in order to optimise the success of the neural network. Although the dataset is mostly complete, there are a small number of instances where data is missing due to sensor problems or similar issues. For these cases, the value from the previous hour has been used on the assumption that the weather will not vary to a huge degree within such a short time period. This patching of the incomplete dataset will mean that no days will

```
Station: Logan
Date: 2008/11/01
Precip: 0.0
Temp: 56
Wind Speed: 3
Wind Directon: 150
Wet Temp: 50
Dry Temp: 56
Altimeter: 30.32
Dewpoint: 44
Station Pressure: 25.7
Sea Pressure: 30.22
Humidity: 65
```

Figure 9 - example of NCEI weather data from one hour of a day - [31]

be lost or unable to be utilised. Although this method adds a degree of inaccuracy to the data, it will improve the ability for the system to learn from previous days leading up to an avalanche.

The data collected by UAC about avalanches is highly detailed, including a large amount of information regarding the type and size of each avalanche. Unfortunately, it is not always possible to collect complete information about each avalanche as many of the recorded instances have been reported by members of the public [32]. For this reason, only variables which are commonly complete have been extracted from the dataset. An example of the variables which have been extracted can be seen in Figure 10.

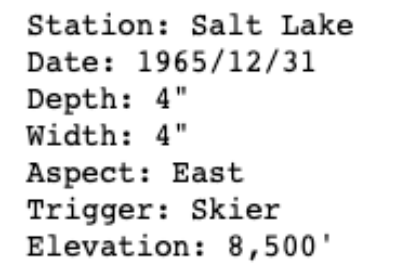

Figure 10 - example of UAC avalanche data from one day - [32]

Successive implementation methods will aim to achieve a more sophisticated prediction system. Initial methods will make predictions based upon whether an avalanche is or is not likely to occur. Within these methods, most of the variables within this dataset will not be used, however more advanced systems may attempt to predict the magnitude or terrain in which an avalanche may occur.

Although the avalanche dataset's earliest entry is from 1914, there are very few avalanches recorded until the mid 2000s. The first winter season to contain recordings at regular intervals is 2008/2009 and from the 2009/2010 winter season, recordings are highly consistent. For this reason, data preceding the 2008/2009 season will be disregarded. Data recordings from this point onwards will be used and the implementation methods within this project will explore the suitability of the remaining data.

Although weather data is available for 365 days per year, only the data during the winter season is useful for the application of avalanche prediction. The earliest month in which avalanches are recorded is December. However, the weather previous to these avalanches must be considered as an avalanche contributor. Therefore, November has been chosen as the start of the relevant winter data. The last avalanches to be recorded within a year are in the month of May. For this reason, May has been chosen as the final month of the winter season. All weather data outside of this range has been removed as it is not relevant to this project.

3.2 DATA VISUALISATION

3.2.1 PRECIPITATION

For the Utah area, the magnitude of precipitation across a year is relatively even. Figure 11 shows the amount of precipitation per day, for the days which precipitation occurred. Each red line shows the separation of each winter season and the blue datapoints within each of these sections denote the number of inches of precipitation which occurred in one day. However, the figure below only displays the days on which precipitation did occur. Such days totalled to 628 across the 10 seasons at Logan, however there were 1494 days where precipitation did not occur. In contrast, the Logan region reported a total of 931 avalanche days across the 10 year period. Although the presence of precipitation can be attributed to a higher risk of avalanche, this relationship does not form a linear correlation. Instead, the neural network must rely on additional parameters in order to make its predictions.

In addition to the concentration of precipitation days, the amount of precipitation which occurred plays largely into the formation of avalanches. As seen in Figure 11, the median value for precipitation lies between 0.01 and 0.15 inches. However the range of this data is from 0.0 to 0.8 inches. The days which see abnormally high amounts of precipitation will contribute to a high avalanche risk in the following days. The opposite can be said for days with little to no precipitation. When snowfall is low, the following days will have a much lower chance of avalanche and this effect is exaggerated the more time has passed since the last snowfall.

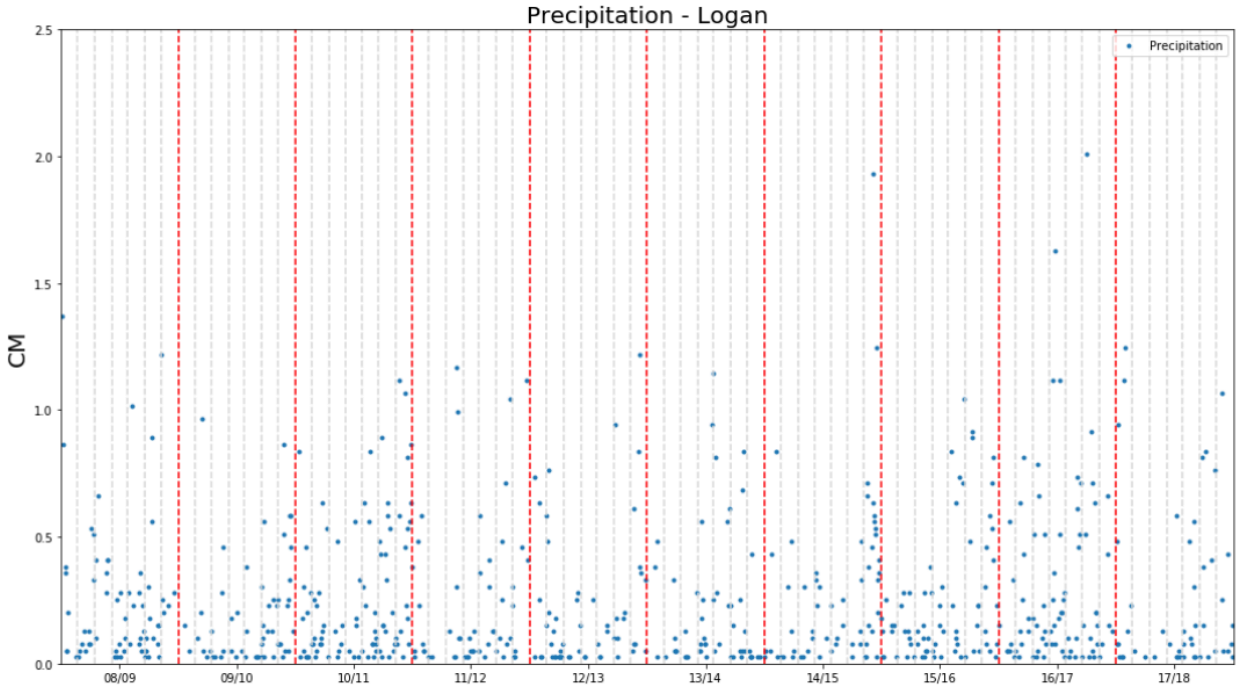

Figure 11- 2008-2019 Precipitation readings from Logan, UT

3.2.2 TEMPERATURE

As seen in Figure 12, temperature across a winter season follows a distinct trend. During the early months, temperatures drop at a high rate. During December, temperatures reach some of their lowest of the season, most years dropping to around -20°C. From December to February, temperatures stabilise with little fluctuation or drastic change. This type of weather creates dryer snow with very little thawing of the snowpack, making for a more stable snowpack which is less likely to avalanche.

From March onwards, through spring, the temperatures begin to increase. With this comes wetter, heavier snow and a freeze/thawing effect as day transitions to night. These changes in temperature will cause the formation of snow and therefore the snowpack to vary drastically. Avalanches will occur later in the season differently to the start and the parameters seen to contribute to these will be different too. As detailed earlier in section 2.1 of this paper, there are many different types of avalanche. These are caused by different types of snow within the sliding layer of the avalanche. As snow is formed largely in relation to temperature, the contributing factors to an avalanche will change as the winter season progresses. The neural network will need to understand how avalanches can be formed in different ways throughout the season and use this within its prediction.

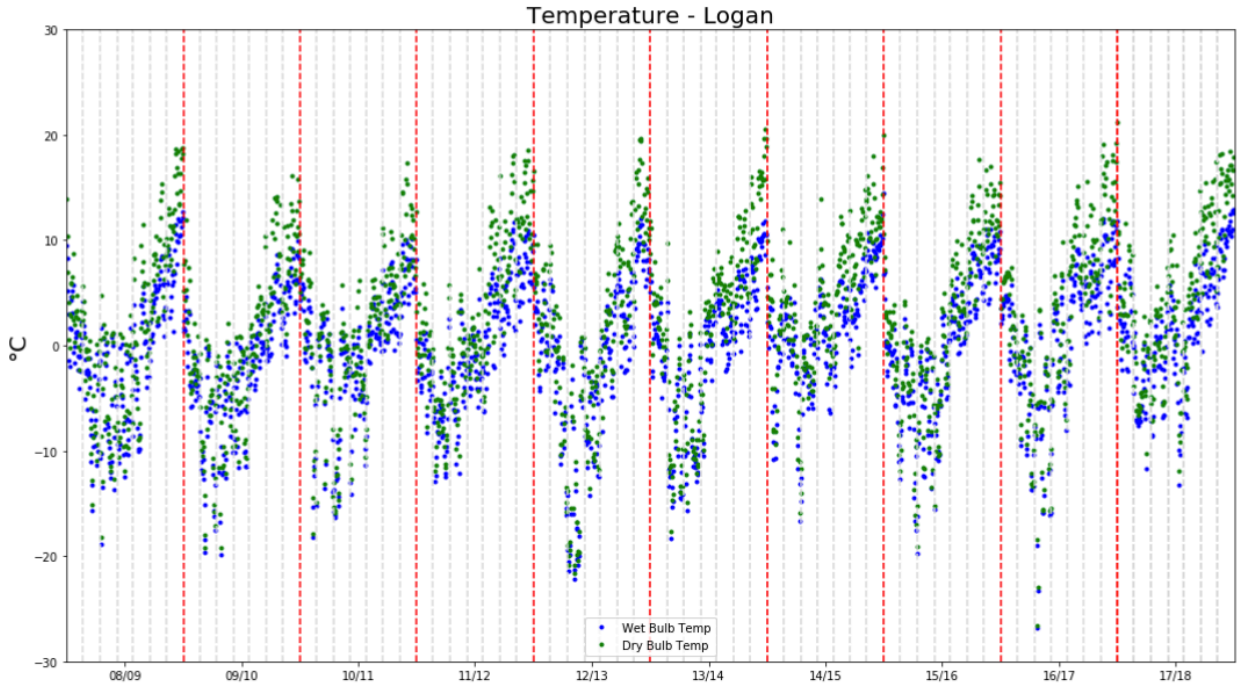

Figure 12- 2008-2019 Wet and Dry Temperature readings from Logan, UT

3.2.3 WIND SPEED

Wind speed is the final of the three variables which contribute the most to the formation of avalanches. Its results can be seen in Figure 13. Throughout the season, wind varies constantly. No obvious trends can be seen in this data as the season progresses and the distribution of high and low levels of wind is spread evenly across the winter season. The network will need to understand that higher winds contribute to a higher risk of avalanche. This approach can be used throughout its calculations.

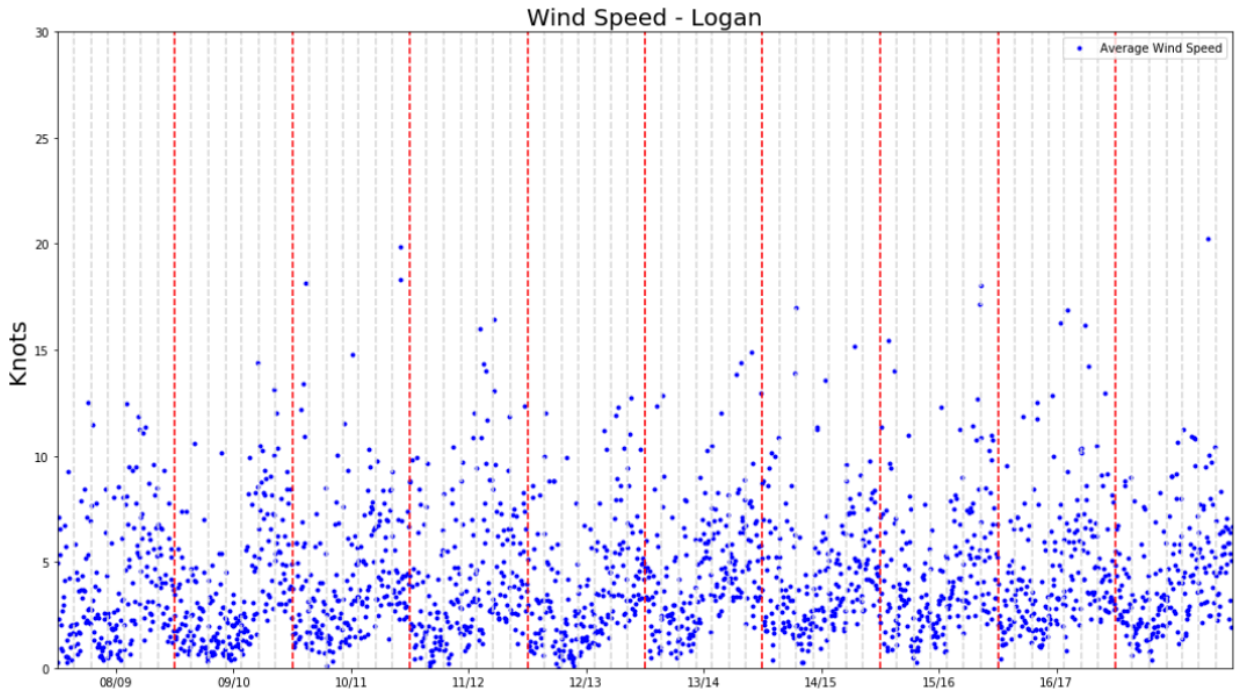

Figure 13- 2008-2019 Wind readings from Logan, UT

3.3 DATASET LIMITATIONS

Due to using real world data for this project, there are a number a limitations to the project's success that arise from the dataset itself. The first of these to arise is the accuracy of the data. Although measured to a high precision, it is not possible to know whether the data being recorded by the sensors was influenced by factors outside those that it was intended to measure. For example, wind speed measurements close to physical structures will be influenced by the currents created from air moving around the buildings. This type of interference could impact the accuracy of the data being used in this system. Within the meteorological measurements, these inaccuracies could cause small changes to the data.

In addition to inaccuracies within the dataset, there are some readings from the weather data which are missing entirely. Within the NXD systems, this type of loss caused a chain reaction of problems with their prediction system. As each prediction relied on the three previous days' data, losing information from one day would mean that the system could not use the next three days worth of data within their model. Missing data can cause big problems for machine learning systems and the methods used to counter this are detailed in the following sections of this paper. Furthermore, if an avalanche was not recorded, this could impact the system's ability to correctly learn the contributing factors which lead to avalanches. Not only would this increase the number of True Negatives but also impact the predictions being made following such a missed avalanche.

For a neural network to successfully perform supervised learning, all data must be labeled. Therefore, it is important that the weather data which is fed into the system has corresponding avalanche data supplied alongside. In order to find weather data which mapped directly to the regions detailed within the UAC avalanche data, local airport weather stations were used. Although airport weather stations are renowned for their accurate and comprehensive data, they do not give an accurate indication of the weather which is occurring within the mountains themselves. Although the general weather trends will be captured, the prediction system is limited by its imprecise knowledge of the mountain environment. The methods used to reduce this impact will be detailed in the following sections of this paper.

One problem related specifically to the RNN structure, which will be used in this project, is the reading in of data in a linear fashion across multiple seasons. Because the RNN learns from the data of the current iteration as well as the data preceding it, with early season predictions, the weather from the end of the previous season may be taken into account. This data has no direct effect on the formation of avalanches and may inhibit the system's ability to learn.

3.4 DATASET FORMAT AND PREPROCESSING

3.4.1 DATA STRUCTURE

Before the dataset is fed into a neural network, it must be suitably structured and normalised. This step was completed using Python's Numpy library. Data was read from its original CSV format and stored in a Numpy array. The structure of the weather array was (2122, 24, 9), storing 9 weather variables for each hour within a day, across 2122 days for a single weather station. The number of days was calculated from 10 winter seasons starting 2008/2009 and finishing 2017/2018. A winter season is defined between November 1st and May 31st, totalling 212 days per season. However, two leap years exist between 2008 and 2018, adding an additional 2 days to the total. The structure of the label data array was (2122), simply storing a single boolean value for a positive or negative occurrence of an avalanche on that day.

3.4.2 AUGMENTATION AND CLEANING

On inspection of the data, it found that there were many missing weather readings from the NCEI dataset. In order to minimise the negative effect of missing data on the prediction system, the missing values were replaced. The method for this was to take the value from the previous hour's reading and replace the missing value with this. In addition to missing data within the dataset, there were many readings with erroneous characters inserted. For example, some data readings would contain a falsely placed letter 's' after the value. These were removed while loading the dataset into python variables. These subtle changes ensured that the data being processed by the system was complete and in a standardised format. It also eliminates the need for entire days to be discarded when errors are found in the data.

3.4.3 NORMALISATION

Neural networks attempt to find trends in data which lead to the correct classification of an input. When the data being fed into the network has a large range, it becomes more difficult to find these trends. When data is plot evenly across a small range and variation is reduced, it becomes much simpler to find these trends. For this reason, neural networks work better when data is normalised before it is input. Some values within the NCEI weather dataset span a wide range of negative and positive values, while others are only positive. There are a number of different data normalisation functions which are optimised for different distributions of data.

The normalisation used upon the data for this project may vary through different implementation methods and these will be detailed when applied. Many of these can also skew the importance of certain changes in data within one portion of the distribution. These can be useful with temperature values when the melting point and boiling points are reached, in order to add importance to small changes in these ranges. Figure 14 shows the function of such a normalisation function upon a temperature value approaching 0°C. In addition to normalisation functions, some data may require further modification. In the case of wind direction, the measurements taken are the angle in degrees at which the wind is blowing. As the range of this data is permanently fixed between 0 and 360, the dataset can be scaled by dividing the value angle value by 360. This type of normalisation does not shift the importance of data within the range but it will scale or shift the data in some way. Throughout each implementation method, the success of these variables will be reviewed.

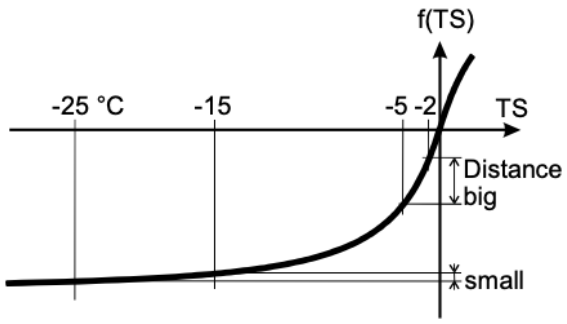

Figure 14- normalisation functions for temperature values - [1]

4.0 EXPERIMENTS

4.1 INTRODUCTION

The following section will provide a detailed description of multiple implementation methods of an RNN based Avalanche Prediction System using LSTM architecture. Methods will utilise different parameters in order to achieve an optimal result. This may include changes to network architecture, data normalisation, optimisation algorithm and loss function. The success of each implementation will be evaluated through three separate methods: Accuracy, Receiver Operating Curve (ROC) and F1 score. For the purposes of evaluating success criteria 2 surrounding current published predictions, a target of 10% above baseline accuracy predictions will be used as a goal for the training process.

4.2 TOOLS AND LIBRARIES

For this project, Python has been chosen as the core programming language. The reason for choosing Python comes from the many useful libraries which are available. These include dataset manipulation, graphing and machine learning libraries which will allow for a robust and professional neural network based program to be built. Jupyter Notebook has been used to write and document code. Its ability to segment code into blocks and run these separately allows for clear debugging and easy changes to programming logic. From within Jupyer Notebook, it is possible to compile Python code as well as output text and graphs.

In order to format and process data, the Numpy library has been used. Numpy allows data to be stored in n-dimensional arrays, making it possible to handle weather data in groups of 24 hour periods. This is necessary because it is not possible to define an avalanche occurrence to less than a 24 hour period. As avalanche occurrences are used as data labels, the information needs to be processed in these groups in order to successfully identify contributory trends.

TensorFlow and Keras libraries have been used in conjunction to implement the machine learning functionality of the system. TensorFlow is a machine learning framework which handles the processing of data using a CPU or GPU. TensorFlow is built specifically to handle data processing through large scale neural networks and allows many different structures to be defined. Keras is used to easily define the parameters being used within a TensorFlow network. It provides a clear and straight forward way of interacting with TensorFlow, making it possible to define the structure of a the neural network line by line.

4.3 EVALUATION METHODS

4.3.1 TRUE & FALSE, POSITIVE & NEGATIVE

When a network classifies data, there are four possible outcomes at to how the success of the prediction. A prediction can either be correctly or incorrectly true or false. The goal of the system is to maximise the number of true positives and negatives while reducing false positive and negatives. These variables are used when calculating Precision and Recall.

True Positive - Expecting a true value - System classified correctly	**False Positive** - Expecting a false value - System classified incorrectly
False Negative - Expecting a false value - System classified incorrectly	**True Negative** - Expecting a false value - System classified correctly

4.3.2 ACCURACY

Accuracy is the measure of whether the predictions made by the system were correct or not. This is calculated as:

$$\text{Eq. 10} \qquad \frac{\text{True Positives} + \text{True Negatives}}{\text{Total}}$$

As the system processes a batch of data, it will make predictions about this data. Those scores are stored as the system updates its weights across a number of epochs. This measurement can be taken on while the system is training or during the testing stage. For each of the methods, both the training and testing accuracies will be considered during evaluation. For datasets which have an uneven balance of classes, it is possible to simply predict the class which contains the largest number of examples. This method will appear to produce a better than random prediction accuracy. However, a system using this method will not learn the trends of the data and will not be useful in categorising data. For this reason, the default (baseline) accuracy will be shown as a comparison to the accuracy of each system. Only systems which perform higher than this level will be considered to be in any way successful.

4.3.3 LOSS

Within a neural network, a loss function is defined. This loss function determines whether the system is successful in order to adjust weights during training [33]. The loss metric is simply the values which are calculated by the loss function. As each batch of data is fed into the network, the loss for each prediction is calculated. The output from this function is then plot against each epoch of training. A low output reflects a small number of mistakes and is therefore desired.

4.3.4 PRECISION

The precision or sensitivity of predictions calculates the percentage at which positive predictions were made correctly [34]. The formula for precision is as follows:

$$\text{Eq. 11} \qquad \frac{\text{True Positives}}{\text{True Positives + False Positives}}$$

Within this system, there are only two classes which are being used: An avalanche occurring or an avalanche not occurring. Precision will give the rate at which avalanche days were successfully predicted against all of the predicted avalanche days.

4.3.5 RECALL

Recall, also known as specificity, is used to calculate the the rate of correct positive predictions against all of the positive data [34]. Recall can be calculated as:

$$\text{Eq. 12} \qquad \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

Within this system, recall will provide the rate that the system predicts an avalanche against every avalanche day that occurred.

4.3.6 ROC/AUC

A ROC (Receiver Operating Characteristics) curve plots the probability of predicting a true positive against a false positive. The Area Under the Curve (AUC) represents how well the system can distinguish between the different classes within the dataset [35]. The higher the AUC, the better the system is performing. An AUC of 0.5 means that the system cannot distinguish between the classes. This will occur when the system simply predicts the most common class every time instead of finding trends within the dataset.

4.4 IMPLEMENTATION 1

The goal of Implementation 1 is to evaluate the performance of different loss functions for avalanche prediction and achieve an initial level of network performance for future implementations to build upon. There are three general categories which loss functions can call into: Regression Loss Functions, Binary Classification Loss Functions and Multi-Class Classification Loss Functions [36]. The avalanche prediction system which will be created in this project uses a binary classification to predict whether an avalanche will or will not occur based upon weather data variables. Therefore, Regression or Multi-Class Classification loss functions will not be appropriate for this task. The classification loss functions which will be explored within this method are:

- Binary Cross-Entropy
- Hinge Loss
- Squared Hinge Loss

4.4.1 MATERIAL

For the first method, a data structure has been chosen which contains 9 decision variables for each hour of a day, for 9 winter seasons, ranging from 09/10 to 17/18. Upon further inspection of the avalanche data, there were substantially less avalanches documented in the 08/09 season than in the following winter seasons. From 09/10 to 17/18, there was a more consistent distribution of avalanche recordings. The reason for this lack of data is most likely to be due to a lack of reporting rather than a year with a much lower frequency of avalanches. In order to reduce the risk of the system learning incorrect trends from this incomplete data, the 08/09 winter season has been omitted. The 9 decision variables being considered are:

- Precipitation
- Wet Bulb Temperature
- Dry Bulb Temperature
- Wind Speed
- Wind Direction
- Dewpoint
- Station Pressure
- Sea Pressure
- Humidity

For this initial build, all variables will be normalised to a range between 0 and 1 by simply dividing each variable by the maximum value within that parameter set. This ensures that the data has consistent limits yet has not been manipulated in any way to shift the importance of particular data ranges.

The datasets have been split into two separate sets; one for training and one for testing. This split has been done at the ratio of 7 seasons : 2 seasons respectfully. The training dataset contains avalanches on 49% of days while the testing dataset contains avalanches on 48% of the days. The baseline accuracies for each of these sets will therefore be 51% and 52% respectively.

4.4.2 METHOD

For the first implementation of an avalanche prediction program, the structure of the neural network has been kept relatively simple. The goal of this implementation is to provide a baseline result set which can be improved upon in further iterations. The structure of the network consists of two hidden layers: an LSTM layer of dimension 200 and a fully connected layer of size 50. The LSTM layer will aim to find trends in recurrent data while the dense layer will find further outline the trends from the LSTM layer. Dropout is used to prevent the overfitting of a network to a training set [37]. Nodes within a neural network build complex relationships between each other. When training the network, these complex relationships are designed to find trends in data. However, catching too many trends could mean that a network becomes overfitted. An initial dropout rate of 0.2 has been defined within the first LSTM layer of this initial model.

The output layer of the network contains only one perceptron and uses a sigmoid activation function. Having a single node in this layer means that the classification of the data will be binary; either predicting that an avalanche will or will no occur. The sigmoid activation function maps values between the range of 0 and 1, as seen in Figure 15. As our model will be predicting an avalanche occurrence as either true or false, this activation function is appropriate.

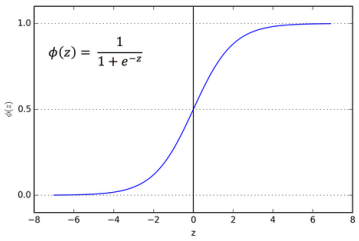

Figure 15 - Sigmoid function - [38]

A batch size of 28 will be used, taking 4 weeks of 7 days at a time. Once one week's data has been read into the network and predictions have been made, the loss of the network is calculated and weights are adjusted accordingly. The program will run for 50 epochs in order to allow the network to fully train on the dataset. This value may be adjusted in later iterations in order to prevent overfitting.

The neural network structure described above, and seen in Figure 16, will be used along with Mean Squared Error, Mean Absolute Error, Binary Cross-Entropy and Hinge loss functions. Binary Cross-Entropy and Hinge are both functions which are suited to binary classification problems. Whereas, Mean Squared Error and Mean Absolute Error are both suited to multi class classification problems [36]. It is expected that the binary classification loss functions will perform better than the multi class loss functions. Separate models will be built with otherwise identical parameters. These models will be referred to as model 1, 2, 3 and 4 respectively. The success of each of these will be measured and evaluated in the following section of this method. The results from this implementation will be used to create higher performance models in subsequent iterations.

```
model = Sequential()
model.add(LSTM(200, dropout=0.2, input_shape=(24, 9)))
model.add(Dense(50, activation="tanh"))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss=loss, optimizer=optimiser, metrics=['accuracy'])
```

Figure 16 - Model definition from method 1

4.4.3 RESULTS

After running each of the models for 50 epochs, the accuracy and loss scores for prediction on both the training and valuation sets were calculated. Each model's performance has been plot on the same graph in order to provide a clear comparison. In order to give a reference of the accuracy performance, the baseline accuracy has been plot in grey. This shows the accuracy which would be achieved from simply predicting the larger class in each classification.

As seen in Figure 17, models 1, 2 and 3 all begin to learn from the training set. Accuracy scores steadily increase from the first epoch, showing that these models are working to some degree. However, around 30 epochs, model 2's accuracy drops sharply to meet the baseline accuracy. From here onwards, this model simply predicts the larger class, failing to successfully learn trends within the data. Models 1 and 3 both continue to improve in accuracy as they progress through additional epochs, reaching a maximum accuracy of around 70%. Model 4 shows a constant accuracy of exactly 47.53 across every epoch. In each prediction, this model is choosing the smaller class and not learning from the data whatsoever. At this stage, models 1, 2 and 3 all appear to show promising results. Model 4 will not be discussed further as it has been deemed to be unsuccessful.

Figure 18 shows the accuracy scores for predictions made on the unseen validation dataset. These accuracy scores give a closer indication to how the system would perform on real world data as the network has not been trained specifically on them. Similarly to what is seen during training, models 1, 2 and 3 start by improving in accuracy. Each model begins at or close to the baseline accuracy and, across the first 20 epochs, improves to an accuracy of close to 60%. After 20 epochs, each of the models begins to drop in validation accuracy. After 30 epochs, each of the models is predicting at a rate equal to or below the baseline accuracy.

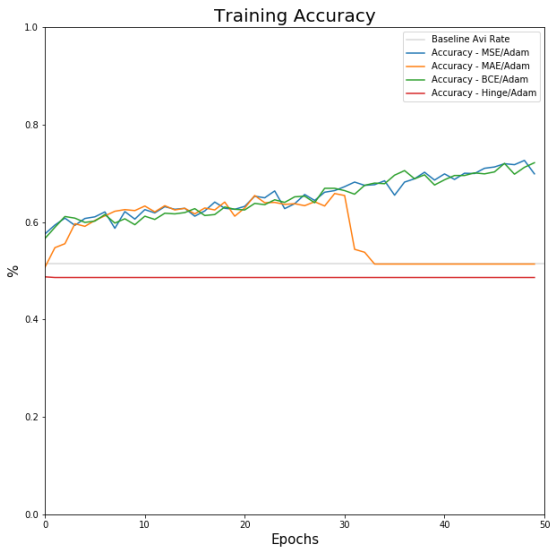

Figure 17 - Training Accuracy

Figure 18 - Testing Accuracy

Loss is the value taken directly from the loss function being used in each model. As each loss function varies in how it calculates its value, the value of each function is not relevant. However, the shape of the graph shows the rate of incorrect predictions and this will be analysed. Figure 19 shows that during training, models 1, 2 and 3 all exhibit a steady reduction in loss. This shows that each of these functions is learning from the training data. At 30 epochs, the loss of model 2 quickly increases and remains at a rate equal to the starting loss level.

As seen in the accuracy results above, Figure 20 shows steady improvement across models 1, 2 and 3 up to 20 epochs. After this level, loss beings to increase, indicating that the models are becoming overfitted to the training set. Each of the graphs discussed show a clear concurrency in that training from 0 to 20 epochs is successful and training past this produces overfitting.

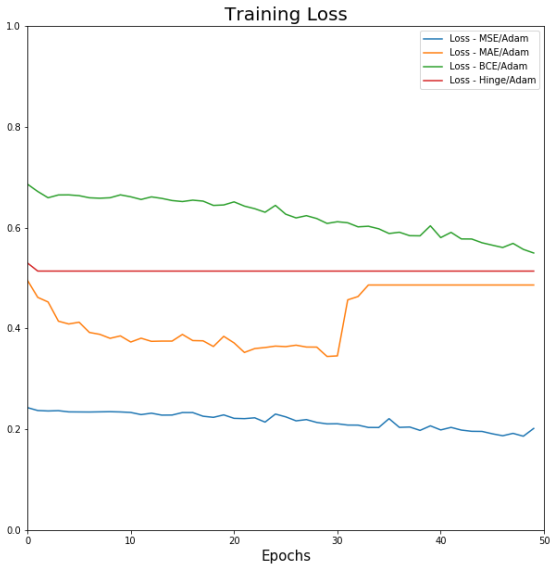

Figure 19 - Impl. 1 Training Loss

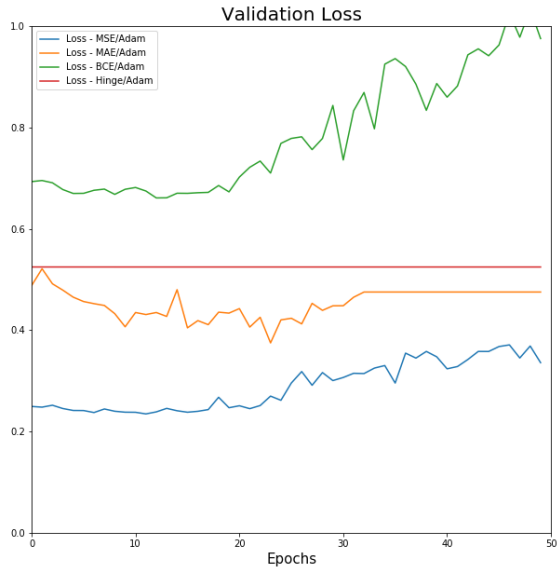

Figure 20 - Impl. 1 Validation Loss

A ROC curve gives an indication as to whether a model has successfully learnt from a dataset or not. A larger Area Under the Curve (AUC) suggests a successful model. An AUC of 0.5 shows that a model has made an equal number of good predictions as bad predictions. Figure 21 shows the ROC curve for each of the models when trained across the first 20 epochs. Instead of plotting this curve for the full 50 epochs which were run in the method above, the model history was cut before overfitting began. These ROC curves show that models 1, 2 and 3 were more successful than random guessing.

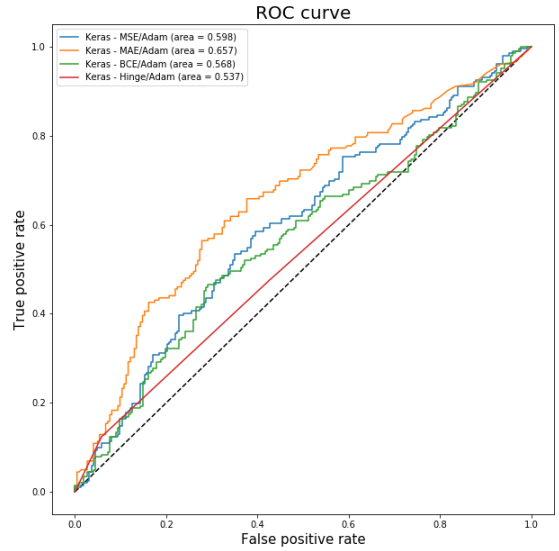

Figure 21 - Impl. 1 ROC across 20 epochs

4.4.4 CONCLUSIONS

The first conclusion that can be made from this implementation method is that the hinge loss function is not compatible with this model. It failed to learn any trends within the dataset and consistently produced accuracies below the baseline accuracy.

Secondly, from each of the evaluation methods above, a clear trend can be seen. Models 1, 2 and 3 were all successful at predicting the occurrence of an avalanche at a rate higher than random guessing. Learning improves across the network for the first 20 epochs, at which point overfitting to the training data begins. The validation accuracy scores for the Binary Cross-Entropy based model were higher than any other model. However, the difference between validation accuracy scores of models 1, 2 and 3 were very small.

Although the model expected to perform the best was able to successfully learn, the two models which were expected to perform worse resulted in similar performance as the Binary Cross-Entropy model. This may be due to having a small amount of data to learn from. With such a short period where the model was predicting correctly with the validation dataset, the difference between each of the models may not have been pronounced enough to compare.

Overfitting is the result of a model learning the trends of the training dataset instead of learning the trends which actually contribute an avalanche occurring. This could be due to the size of the dataset being too small. Larger datasets provide more examples of positive occurrences for the network to learn from [39]. The models trained above used 9 winter seasons of data from the Logan weather station. There are a number of other weather stations available to be used and extending the dataset with these may extend the training which can be done before overfitting begins.

Another method of preventing overfitting is to remove features which are not relevant to the system [39]. The removal of these variables means that the model is less likely to learn incorrect trends in the data. Such variables act as noise and cause distractions in trends.

Cross-validation is the act of segmenting the training and testing datasets into smaller sets and then shifting which parts of the dataset are used for training and which are used for validation [40]. Instead of using the training and testing sets as completely separate datasets, the training and testing sets are combined and these are proportionally segmented into sets, known as folds. One of these folds will be used as the validation set and the others used as the training set. The fold which is used for validation cycles through each possibility. Once this has been completed for each possible combination of folds, average loss and accuracy values are calculated across each iteration.

4.5 IMPLEMENTATION 2

The goal of Implementation 2 is to learn from the results of Implementation 1 and to adapt the network in order to achieve a higher level of validation accuracy and to reduce the impact of overfitting on the models. Once results have been gathered, the performance of this implementation will be compared against those of Implementation 1.

4.5.1 MATERIAL

During Implementation 1, overfitting caused the network to drop in validation accuracy after approximately 20 epochs of training. In order to address this, a larger dataset will be used. The weather data from the Logan and Salt Lake City weather stations will both be used in the training and testing datasets. The shape of the array used to input data into the network will therefore be changed. Additionally, the number of weather variables used will be reduced. Although this reduces the amount of data which the network can learn from, it will remove variables which have a low influence on avalanche occurrence. The variables now fed into the network will be:

- Precipitation
- Wet Bulb Temperature
- Dry Bulb Temperature
- Wind Speed
- Wind Direction
- ~~Dewpoint~~
- ~~Station Pressure~~
- ~~Sea Pressure~~
- ~~Humidity~~

The data will now be stored in an (X, 24, 20) Numpy array for processing. Each day will contain 24 sets of data, with 5 parameters from the Logan, Ogden, Salt Lake and Provo weather stations. This structure will extend the data which the network can learn from but also provide a less localised view of the weather systems affecting the mountains.

4.5.2 METHOD

Further to extending the dataset, a method of preventing overtraining which will be used in this implementation is Cross-Validation. The 9 seasons of weather data will be split into 9 folds, each of one season in length. The network will be trained using each of the 9 folds successively as a validation sets. The training and testing accuracy and loss will be calculated for each and averaged. This will provide a larger dataset for the network to train on while also displaying the network performance when trained on varying datasets. This reduces the impact of a single data fold containing data which lies outside of the general trend.
As in Implementation 1, the same general model will be used with a number of different loss functions. The functions used will be Mean-Absolute Error, Mean-Squared Error and Binary Cross-Entropy. These three loss functions performed well previously and were not easily distinguishable in performance. The averaged results of cross validation for each of the models will be plot and compared.

In order to allow the network to fully train on the larger dataset, 100 epochs of training will be carried out for each model. This extended training period will show whether the new models are able to successfully learn from the dataset and at which point overfitting begins, should it happen at all. The structure of the models used will be the same as in Implementation 1 and this can be seen below in Figure 22. This structure has remained consistent in order to limit the number of changes made between implementations.

```
model = Sequential()
model.add(LSTM(200, dropout=0.2, input_shape=(24, 9)))
model.add(Dense(50, activation="tanh"))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss=loss, optimizer=optimiser, metrics=['accuracy'])
```

Figure 22 - Model definition from method 1

4.5.3 RESULTS

After the above detailed changes were applied to the network and the training was complete, the results were plot. The networks tested in this implementation used three different loss functions: Mean Squared Error, Mean Absolute Error and Binary Cross-Entropy. These models will be referred to as model 1, 2 and 3 respectively. The results of Implementation 2 will be compared against those from Implementation 1. The purpose of this is to provide perspective into the progressive performance of the models.

Figure 23 shows the average training accuracy of the three networks across across 50 epochs using 9 fold cross validation. Models 1 and 3 both performed well during training, increasing steadily in accuracy. Model 2 reached a maximum accuracy of just over 60% and then reduced slowly. This shows that model 2 was not able to successfully learn from the training data. During Implementation 1, the accuracy of this model dropped to baseline accuracy and did not progress after overfitting. The accuracy from Implementation 2 shows that this problem was not encountered.

During validation, the accuracy of predictions for all three of the models was very similar. Each model was able to consistently predict at a rate higher than the baseline accuracy, reaching a rate of 60% or higher during the first 20 epochs and slowly reducing over time. This trend can be seen in Figure 23. The accuracy results from Implementation 2 differ greatly from those of Implementation 1, where the validation accuracy dropped suddenly after its optimal training period. This result shows that the networks did not become overfitted to the training set as quickly, indicating that the adjustments made to the networks have resulted in positive impacts to performance.

Figure 23 - Impl. 2 Training Accuracy

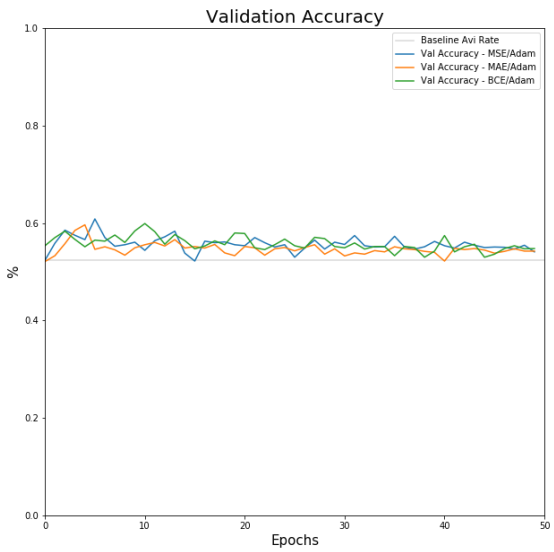

Figure 24 - Impl. 2 Training Accuracy

The loss outputs across the 50 training epochs can be seen in Figure 25. Model 1 and 3 both show consistent improvement throughout training with model 2 learning initially but quickly levelling out to a constant rate. In contrast, the validation loss of models 1 and 3 show the opposite trend, increasing throughout. Unlike during Implementation 1, an initial drop in validation loss cannot be seen. Model 2 shows the same trend in both training and testing. The relationship between training and validation loss for models 1, 2 and 3 is similar to what was seen in Implementation 1, with less extreme features.

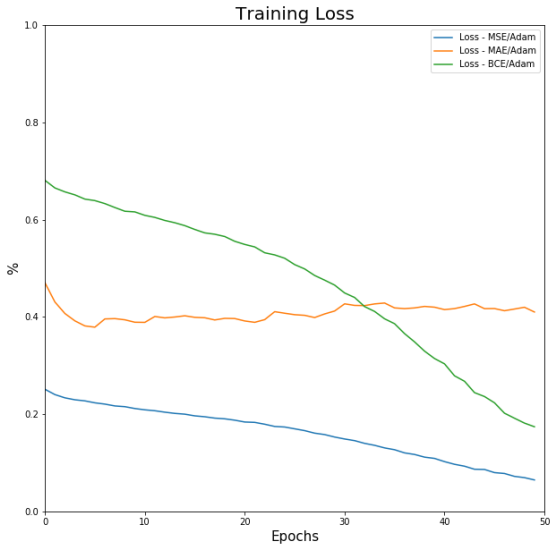

Figure 25 - Impl. 2 ROC Curve

Figure 26 - Impl. 2 ROC Curve

The ROC curve for models 1, 2 and 3 of this implementation have shown the largest difference in performance statistics from implementation 1. These can be seen in Figure 27. The AUC for model 1 averages 0.49 showing that it did not successfully distinguish between classes. Model 2 has an AUC of exactly 0.5, with negligible fluctuation. However, the ROC curve for model 3 clearly shows that learning was successful, achieving an AUC of 0.67. This is the largest AUC achieved by any model from either implementation. Model 3 has shown to be highly successful in all tests carried out within this implementation.

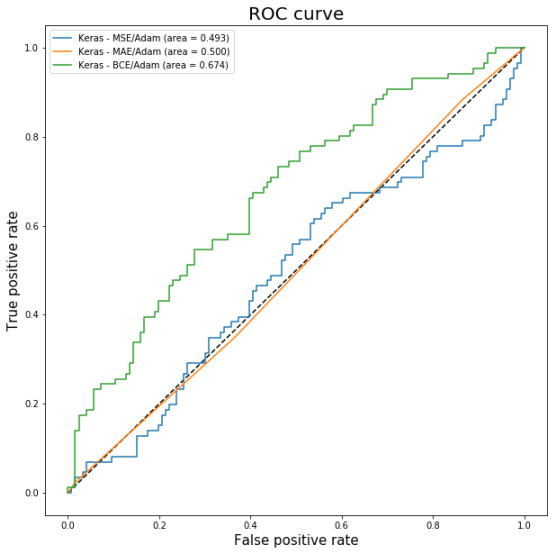

Figure 27 - Impl. 2 ROC Curve

In order to further understand the training of the most successful model, the average validation accuracy across three folds at a time have been plot in Figure 28. This shows how the performance of the model changes as the training data being used becomes more recent. This is a problem which could be seen during training, where the accuracy scores of the earlier folds were much lower than those during later folds. This is likely due to the precision and completeness of the avalanche datasets improving over time. With better data, the models are able to predict at a much higher rate. During the final three folds of cross validation, the average validation accuracy rose above the target accuracy, 10% higher than baseline accuracy, a number of times.

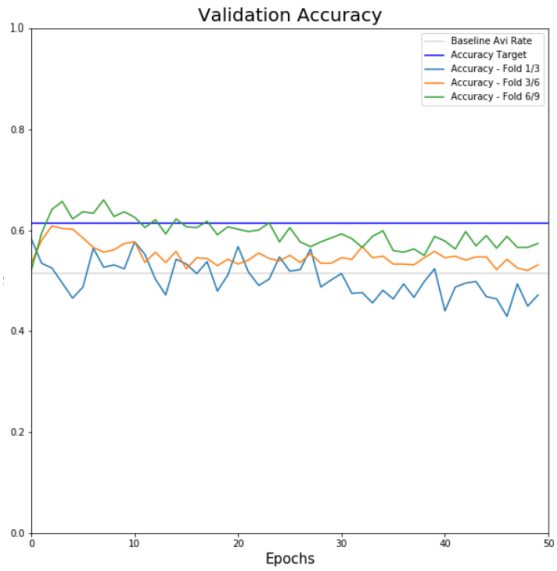

Figure 28 - Impl. 2 - 3 fold validation across 9 folds

Model 3 has proven to achieve a high performance at classifying avalanches from meteorological data, producing a larger AUC value than seen in any other model tested, a consistently positive validation accuracy which, for use upon more complete data, predicts at a rate higher than 10% above baseline. Binary Cross-Entropy is the loss function which was expected to produce the best results and has achieved this across both implementations. Network layer structure and optimiser parameters could be explored to extend the success of this model.

4.5.4 CONCLUSIONS

It is clear that the increase to the dataset has heavily reduced the impact of overfitting of the networks. Each of the models was able to train for over 50 epochs without the validation accuracy dropping below the baseline accuracy. A validation accuracy close to 60% was maintained for a longer period, indicating that the models were able to learn trends within the dataset that contributed to the formation of avalanches. The two largest contributors to this have been the addition of 3 further weather stations to the X dataset and the removal of less relevant data parameters from the dataset. Both of these changes made it easier for the models to spot the trends in weather which lead to avalanches.

With all of the models, the problem of overfitting was greatly reduced and the validation accuracy was improved. The goal for this implementation was therefore successfully achieved. Although improvements were made, not all of the models were successful in their performance. Models 1 and 2 produced an AUC of 0.49 and 0.5 respectively. These results indicate models which are unable to correctly distinguish between each classes and therefore could not learn the trends responsible for avalanche formation. These models will not be carried forward to further implementation methods. The model which was expected to perform best achieved an AUC of 0.67, the highest result achieved in across either implementation. This model shows successful classification and an improved accuracy.

Cross validation has provided much smoother and less extreme curves across each of the performance graphs. Averaging the results from 9 iterations of training means that the best performance is not seen however the results obtained provide a more precise representation of how the model would perform in a real world scenario.

After breaking apart the validation accuracy results across three folds of cross validation at a time, it is clear that the prediction performance is higher for networks tested on more recent data. This is most likely due to more recent avalanche data being more precise and complete. The seasons selected for training were constrained to those from 2009 onwards. This was done for the because the there were a substantially less avalanches recorded before this period and the data was often missing. Since 2009, the UAC have implemented a number of new techniques for recording avalanches such as live public submissions [41]. Such techniques have resulted in a more complete and accurate avalanche dataset.

The final parameter which has not yet been tested is the optimiser function. A range of these could be tested to ensure that the best performing optimiser function is being used the prediction system.

4.4 IMPLEMENTATION 3

The goal of this implementation is to test a number of optimiser functions for use with the current best performing model. The optimisers tested will be the Adam optimiser, one of the most popular optimisers for neural networks having performed top in a number of common comparison tests [42]. Another optimiser which will be tested is Stochastic Gradient Decent. This is one of the first optimisers to show highly reliable performance. Finally, RMSprop will be used, which is known to perform well with RNN architectures [43].

4.4.1 DATASET

The dataset structure for this Implementation will be identical to that of Implementation 2. This will ensure that the optimiser functions are fairly compared. 9 fold cross-validation will again be applied to each of the models and the average scores taken. No changes will be made to the normalisation of data or the variables chosen.

4.4.2 STRUCTURE

The neural network architecture will remain identical to model 3 from Implementation 1. The only change will be the optimiser functions which are being tested. This will ensure that the models can be directly compared on their optimiser without any other variables effecting performance. The models using the Adam, Stochastic Gradient Decent and RMSprop optimisers will be referred to as model 1, 2 and 3 respectively.

4.4.3 RESULTS

Figure 29 shows that both models 1 and 3 were able to successfully learn from the training set. They were able to achieve a steady increase in prediction accuracy across the training period. However, model 2 showed a very slow increase, remaining close to the baseline accuracy throughout. The results of validation accuracy can be seen in Figure 30. Models 1 and 3 again performed well, predicting well above the baseline accuracy. Model 1 performed the best out of the three, fluctuating around 58% validation accuracy for the duration of the epochs. Model 3 has a good initial validation accuracy but this rate begins to decline after approximately 20 epochs. This indicates that the model became overfitted to the training dataset much earlier than model 1. Model 2 again shows signs that it was unable to learn from the data. Its validation accuracy fluctuates around baseline accuracy for the duration of the epochs.

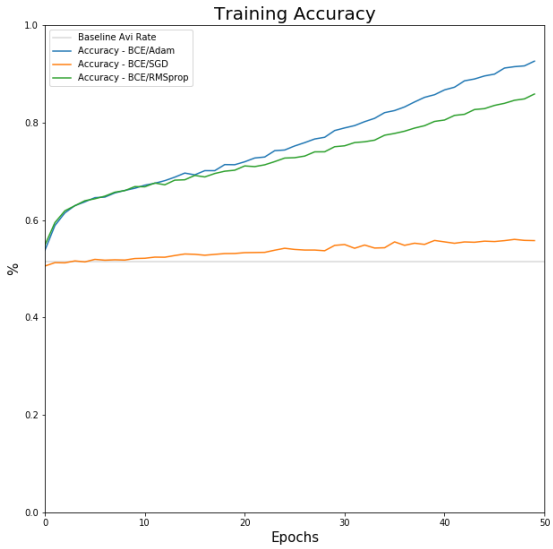

Figure 29 - Impl. 3 Training Accuracy

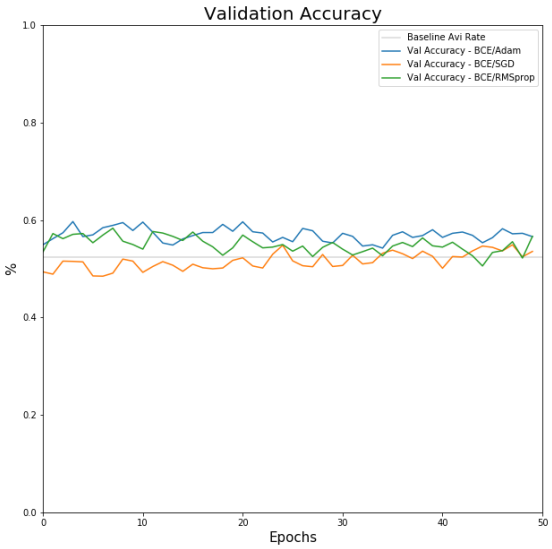

Figure 30 - Impl. 3 Validation Accuracy

The loss functions across training and loss can be seen in Figure 31 and 32. The loss functions for model 2 confirm that it was unable to learn from the dataset. Model 2 will no longer be discussed in this implementation. Loss throughout training for model 1 and 2 showed steady signs of falling. Again, model 1 has produced a steeper curve, indicating that it was able to learn faster than model 3. For models 1 and 3, the validation loss initially drops and then quickly rises. After 10 epochs, the loss value is higher than random prediction. This indicates that overfitting is occurring in both of these models.

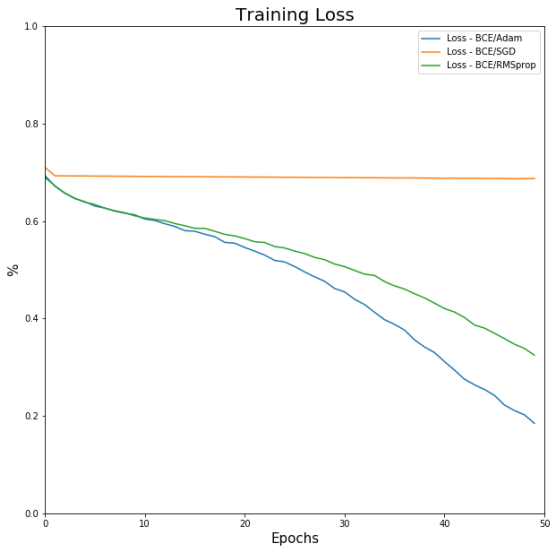

Figure 31 - Impl. 3 Training Loss

Figure 32 - Impl. 3 Validation Loss

The ROC curves for each of the models can be seen in Figure 33. Model 1 shows a similarly positive result as in Implementation 2, achieving an AUC of 0.65. Models 2 and 3 have resulted in similar AUC values, separated by only 0.03. Although model 3 produced positive training and validation accuracy scores, the ROC curve indicates that it has not performed at a level considerably higher than random guessing. Model 2 fluctuates above and below the 0.5 AUC level. It has achieved a score higher than 0.5 however this is likely due to the number of epochs being too small to achieve an even balance of predictions.

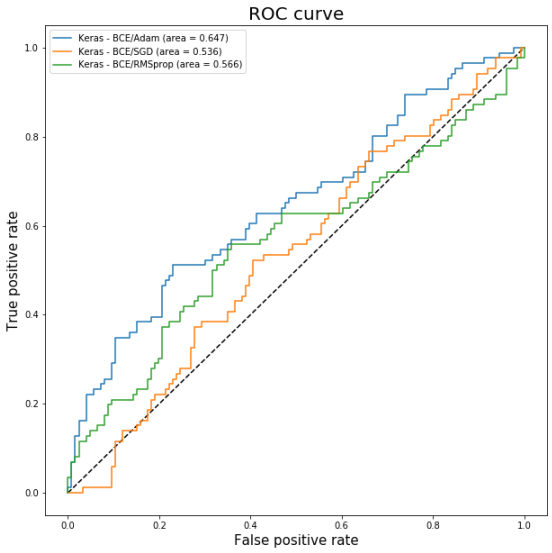

Figure 33 - Impl. 3 ROC Curve

4.4.4 CONCLUSIONS

After testing the three optimisation functions with the highest performing model from implementation 1, clear differences could be seen. The Stochastic Gradient Decent optimiser completely failed to learn from the data. Accuracy with both the training and validation sets fluctuated around the baseline accuracy. The results from the loss functions showed that error of the network was constant throughout training, further solidifying the accuracy results.

Both the Adam and RMSprop optimisers showed a positive prediction accuracies on both the training and validation sets. However, the Adam optimiser performed better in both of these datasets. When inspecting the ROC curves for each of the models, the difference in performance between the Adam and RMSprop optimiser becomes more clear. The ability for model 1 to distinguish between positive and negative classes was much greater than model 3. The AUC of model 1 was 0.65, only 0.02 from the highest AUC performance across the three Implementations. Model 3 however, produced an AUC of 0.56, scoring worse than models 1, 2 and 3 from Implementation 1. This rate close to 0.5 indicates that the network was not good at distinguishing between classes and was not successful.

It is clear that the Adam optimiser paired with a Binary Cross-Entropy loss function has performed to the highest standard of any model across each Implementation. These parameters should be considered when further optimising such an avalanche prediction system.

5.0 REVIEW

5.1 SUMMARY

After conducting extensive research into avalanche science and their prediction methods, it was clear that this field is largely supported by a small number of experts, conducting complicated tests and calculations to gauge avalanche risk. Although these methods are time consuming and dangerous, they are highly accurate. Between 2006 and 2016 there were approximately 100 reported deaths each year in the Alps [44]. As avalanche predictors publish a danger scale, their goal is to minimise the number of human triggered avalanches and casualties from them. Therefore, there are no published statistics for human prediction accuracy which can be directly compared to any artificial prediction systems.

SLF were the first organisation to professionally implement a machine learning based avalanche prediction system, using a K-NN architecture. In 2002, SLF published a paper documenting that their network was able to match the predictions of their official forecasted hazard level at a 52% accuracy [1]. An SLF forecast predicts an avalanche hazard level between 1 and 5, making their model a multi-class classification problem. This model is the most advanced avalanche prediction system currently working in real mountain resorts. Since its development and initial rollout, this system has undoubtedly improved in the accuracy of its predictions. However, the current accuracy results are not publicly available for more up to date systems.

An LSTM Recurrent Neural Network was built as part of this project with a goal of predicting the occurrence of an avalanche on a given day based upon weather data at a validation accuracy 10% above the baseline accuracy. Three implementation methods assessed the change of a single element of the system. Implementation 1 compared different loss functions against each other. Hinge loss failed to learn from the data while Binary Cross-Entropy, Mean-Squared Error and Mean-Absolute Error successfully predicted with a validation accuracy higher than the baseline accuracy of 52%. However, after only 20 epochs of training, the networks all become overfitted to the training set and the validation accuracy dropped below baseline accuracy. The AUC values for models 1, 2 and 3 tested in this implementation ranged between 0.56 and 0.65. These show that the network was able to distinguish between the different classes successfully.

In order to improve the performance of this system, the dataset was extended to include an additional 4 weather stations. This reduced the effect of overfitting, meaning that the networks were able to train for over 40 epochs before dipping below baseline accuracy. Also, 9 fold cross-validation was used to average the metrics taken from the networks in order to reduce the impact of anomalous results. The maximum accuracy of the predictions in this implementation extended above 60% validation accuracy. Model 3 produced an AUC of 0.67, the highest of any model up to this point.

In order to further understand the performance of the network across the different folds of cross validation, the average results from three folds were plot. This showed clearly that the accuracy of predictions improved with use upon the more recent data sets. The cause of this is likely to be due to the accuracy and precision of avalanche results recorded more recently. UAC have implemented a number of user submission options for avalanche reporting, improving the precision of avalanche recordings. The predictions made in the final three folds of cross validation remained above 62% validation accuracy for the first 20 epochs of training. This reaches the target of the project for predicting at 10% above baseline accuracy. This shows that the model built within this project was successful in its implementation. In order to achieve this level of accuracy across full cross-validation, a more complete modern dataset should be used.

A final Implementation method was carried out in order to evaluate the performance of different optimisation functions. Two new functions were tested against the Adam optimiser which had been used previously. It was found that only the Adam optimiser created the best performing model. A binary cross-entropy loss function and Adam optimiser should be used for any further work upon this topic.

The best performing model from this project achieved a maximum validation accuracy of 66% and an average accuracy of 52% with 9 fold cross-validation for 9 years of data. This project's system uses a binary classification problem of whether an avalanche will or will not occur on a given day. SLF have produced their system to predict 1 of 5 separate classes, making their baseline accuracy considerably lower and their problem much harder to solve than the problem addressed by this project. Predictions made by the NXD systems were averaged at a 52% accuracy and achieved a maximum of 61% across 10 fold cross-validation for 10 years of data [1]. The predictions made by the system developed in this project produce outputs with a positive accuracy, considerably higher than a rate of random guessing. With future improvements detailed in the following sections, an LSTM avalanche prediction system has the potential to reach the relative accuracy improvements achieved by the NXD systems.

5.2 CONCLUSIONS

The success of this project is determined by three objectives. First, extensive research was carried out in the field of avalanche science and associated prediction methods. 10 years of weather and avalanche data was collected for the North Utah mountain region. A neural network was then designed and built. Based upon an LSTM architecture, it was able to distinguish weather data which contributed to the formation of avalanches. The baseline accuracy of this model was 52%. It was able to predict at a maximum recorded validation accuracy of over 10% higher than baseline accuracy with an average validation accuracy of 58%. Finally, the system was compared to and evaluated against the current predictive methods. In conclusion, the success criteria of this project have been met and the project has been an overall success.

As the safety of all mountain users is impacted by the avalanche report and risk levels, it is imperative that the systems used by experts to support their predictions must be highly accurate and reliable. The system built within this project does not perform at a level high enough to be used professionally. However, the application of LSTM neural networks has proven to be successful in predicting avalanches from weather data. This project has shown that these networks have the potential to improve further and make predictions at a level suitable for professional application, as shown through comparison to the NXD systems developed by SLF.

In conclusion, the most successful model produced within this project used the Adam optimiser with a Binary Cross-Entropy loss function. These were selected as they produced the highest performance across validation accuracy and AUC results when compared to other parameter selection. Two hidden layers were used including an LSTM layer of size 200 and a fully-connected layer of size 50. The LSTM layer provided the ability to learn from successive sequences of data, while the fully-connected layer extracted the relevant features from the LSTM output.

The largest improvement to the accuracy of predictions made within this project was to the dataset being used. The addition of more weather stations allowed the network to prevent overfitting of the training set, while the use of more recent and complete avalanche data increased the accuracy of predictions more than any other change. With with the use of more detailed and precise avalanche and weather recordings, such machine learning systems will be able to make much better predictions than is currently possible.

Although this system does not perform to a high enough level to replace or support avalanche forecasters, it does show that neural networks and LSTM architectures can be used to analyse trends in weather data to predict the formation of avalanches. Much further research into this topic needs to be undertaken before a functional model could be used in real world examples. However, this project has shed light into a new method of machine learning based avalanche prediction which has not been previously documented.

5.3 FUTURE IMPROVEMENTS

One of the greatest limitations to the prediction system developed in this project was the dataset used. Although meteorological data was precise and spanned a large period of time, the recordings were taken from local airports rather than readings from a mountain location. This meant that the weather which was used to make predictions did not provide an accurate representation of the weather which the avalanche zones were exposed to. The temperature of snow is one of the largest contributing factors to its structure and the properties it holds. Therefore, weather readings taken within mountain regions would provide a more accurate view of the factors contributing to avalanches. The ability for the neural network to learn from these trends would vastly improve its prediction accuracy. This is one area which could make a large improvement to the performance of the models developed in this project.

Furthermore, an extension to the variables available could help to improve trend analysis. Measurements of snowpack compression and snow type are used in most avalanche predictions in order to understand more complicated parameters which are effecting the snowpack. Such variables would provide greater detail for a neural network to learn from, extending its ability to understand trends in data. For example, the NXD-2000 uses a parameter named Settlement (*St*) which describes the way in which the snow has changed over time since it fell. It is calculated using the following formula, where *HS* is the height of the snow and *NS* is the amount of new snow which has fallen [16].

Eq. 2 $$St = HS_0 - HS_{-1} + NS_0$$

These variables could be very useful in a neural network avalanche prediction system but were not possible for use within this project due to the datasets available.

In order to make better use of the datasets which were available, improvements to data normalisation could be made. Compressive normalisation functions such as Sigmoid or Tanh could be used to shift the sensitivity of particular data ranges in order to weight their significance. This would be most useful for variables which have critical transition points such as water freezing at 0°C. This would reduce the amount of unnecessary or irrelevant trends within the dataset, making it easier for a neural network to learn from. The NXD-2000 uses a hyperbolic function to achieve the desired transformation of temperatures. This is defined using the following function where TS is the snow temperature [16].

Eq. 1 $$TS' = 20*\tanh(0.2 * TS)$$

Such functions could be explored in order to further optimise the performance of the neural network.

In addition to changes in dataset and variables used, simply enlarging the dataset available for training and testing would improve performance. It is clear that increasing the number of weather stations being considered has helped to improve trend analysis of the network. If more weather stations were available, it is likely that this improvement would continue. It is important to note that weather data from outside of a localised region could in fact reduce network performance. The addition of data which is not relevant to the predictions being made can cause the network to overfit to the dataset, instead of learning the trends which actually contribute to avalanche occurrence. One way to ensure that this does not happen is by extending the number of years in which the network can learn from. Not only does this ensure that the data remains relevant to the localised area, it also provides more varied examples of avalanches. As avalanches can form from many different types of condition, a longer duration of data would provide more examples to learn from.

Further to dataset based changes, adjustments to neural network structure could be explored. Within this project, Keras was used to abstract the neural network architecture. If raw TensorFlow code is used, a much higher degree of parameter adjustment is available. The LSTM layer within TensorFlow contains 23 different arguments, including parameters related to activation function, bias selection, dropout and return sequences. Due to project constraints, all of these have been unchanged from their default settings within Keras. The optimisation of these parameters would be a large avenue of future improvement for such prediction systems.

The layer architecture within the neural network has also remained unchanged within this project. The implementation of additional LSTM or Fully-Connected layers could influence the network's ability to successfully learn relevant trends from the dataset. Deeper networks tend to be used when the specificity of features being detected needs to be increased. Additional layers will look at the trends made from the individual features extracted from previous layers. This tends to be useful when trends are very complex and a single layer cannot enough relevant information. If more data parameters were added to the dataset, this could help to improve the network performance. However, with only temperature, wind speed, wind direction and precipitation being used, multiple hidden layers may in fact obfuscate the important trends in the data, resulting in reduced performance.

A good improvement to the system design would be to increase the prediction variables which the network is attempting to classify. Although a multi-class classification problem is much more difficult than a binary classification problem, it would provide much more useful outputs. For example, a system which was able to predict not only whether an avalanche would occur but also at what aspect, altitude or snow type an avalanche would have. This extension would indicate which areas of a mountain were high risk and which were not. These areas could then be closely checked by avalanche professionals for a final risk analysis.

5.4 STATEMENT OF ETHICS

This goal of this project has been to research and develop a system which is able to predict avalanches. One of the largest benefits of such a tool would be to reduce the amount of time spent in dangerous avalanche terrain by field testers. Furthermore, a better understanding of avalanche probability would help to provide improved avalanche control as well as allowing for the early implementation of safety mechanisms to protect mountain users from the dangers of avalanches. A successful prediction system would help to reduce the risk taken by mountain users and minimise the number of unnecessary fatalities seen each year. However, a bad prediction system has the potential to put mountain users in greater risk. False negative predictions could lead to the underestimation of an avalanche risk. Such a result could lead to an increase in human triggered avalanches and therefore an increase in fatalities. The benefits of a successful system are clear but the drawbacks of an unsuccessful system could be catastrophic. Therefore, it is advised that such a system be used only to supplement the predictions made by field experts. Such a tool should be used during avalanche risk analysis to augment the available data rather than replacing any current prediction methods.

The data used within this prediction system was provided freely by two separate sources: the Utah Avalanche Centre and the National Centre for Environmental Information. UAC provided their entire archive of avalanche recordings from 1914 to present. This dataset is publicly available and contains no personally identifiable information or sensitive data. Only statistical information regarding different occurrences of avalanches is stored in this file. The dataset provided by NCEI is also publicly available, containing only historical meteorological data [31]. Again, no sensitive or personally identifiable information is stored in this dataset. No data has been used regarding any individual or their behaviours within this project. As both datasets are publicly available and contain no sensitive information, there has been no need to address GDPR issues.

The system built within this project is designed to accept weather data and make predictions about whether an avalanche is likely to occur on a particular day. The data being fed into the system should be accurate meteorological data. In the event of changes being made to the data, the predictions being made will be effected. It is not possible for an attacker to insert malicious data into the network in order to cause harm to the system. However, if this system was used for live avalanche prediction, the addition of false information could lead to inaccurate predictions and therefore inadequate risk mitigation techniques. In order to successfully do this, an attacker would need to intercept the live weather data being fed into the network and manipulate it before it was used. This is highly unlikely and such security concerns would need to be considered by an organisation which decided to implement such a system.

5.5 PROJECT PLANNING

The timeline of the project was carried out well throughout semester 1. Progress was made into research and review of literature by the Interim discussion in week 8. Data collection began before it was scheduled however this took longer than expected to complete. Collecting a complete dataset for both avalanche and weather data from the same local region was very difficult to acquire. Many organisations were contacted with no response. This stage of the project extended into semester 2. Writing of the literature review started after the interim discussion in week 8. This was also a longer process than expected and also extended into semester 2. Both of these extensions were due to an underestimate in the work required to complete each task.

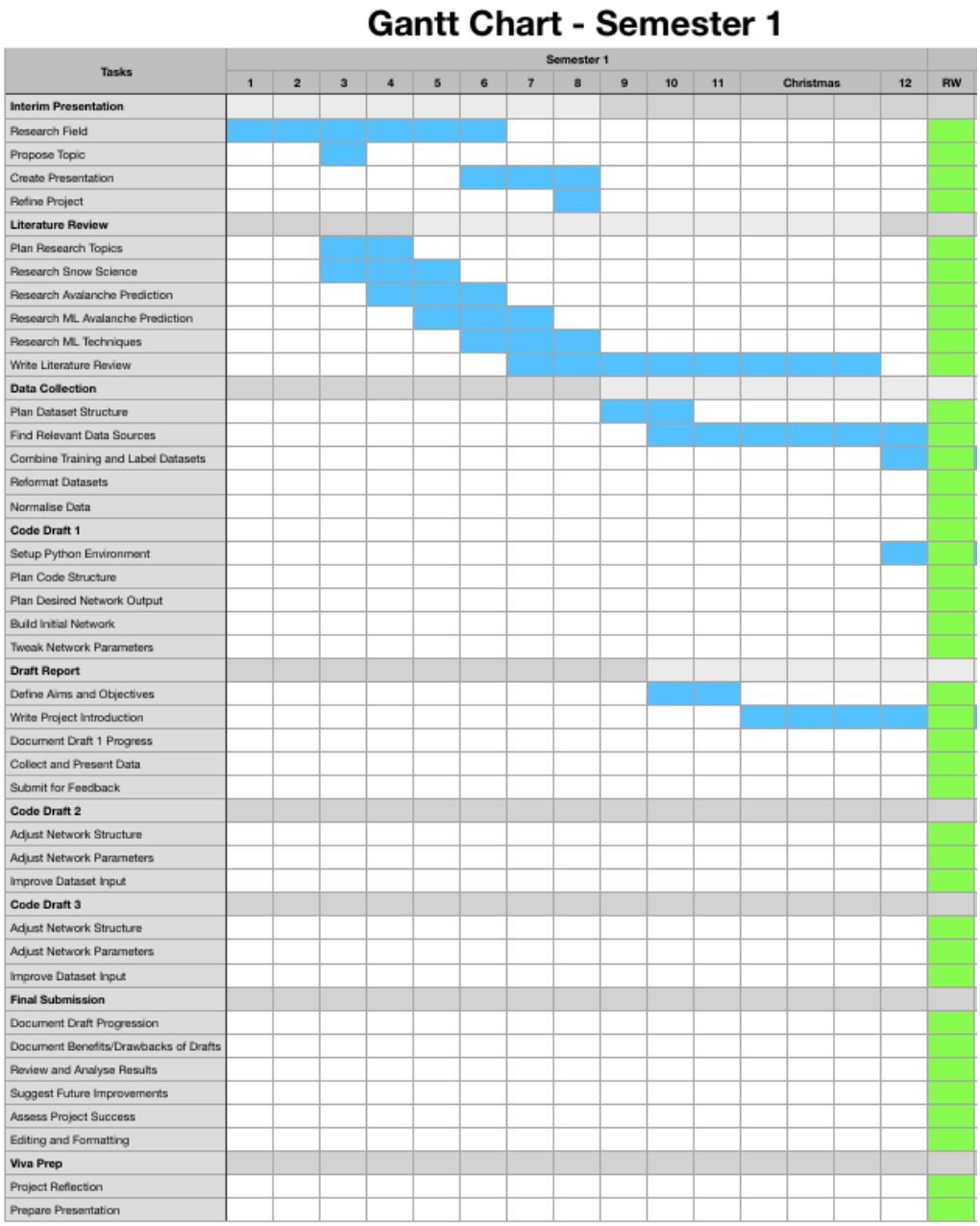

Figure 34 - Gantt chart for Semester 1

Due to tasks allocated to semester 1 overrunning, the tasks planned for semester 2 were delayed. Once the full dataset was acquired in mid February, data formatting and normalisation began. As with many tasks in this project, this took longer than expected to complete. The dataset sourced, although mostly complete, contained many erroneous characters and incomplete fields. Data augmentation and cleaning was required in order to make the dataset usable by a neural network. This unexpected and unplanned section took an additional two weeks, postponing the creation of the first code draft. Once complete, coding the neural network began. Due to the delay in dataset processing, each code draft was completed in less time than allocated. Draft 2 and 3 were both completed within one week each, instead of the three originally allocated. However, by the final submission date each of the sections planned was completed in full.

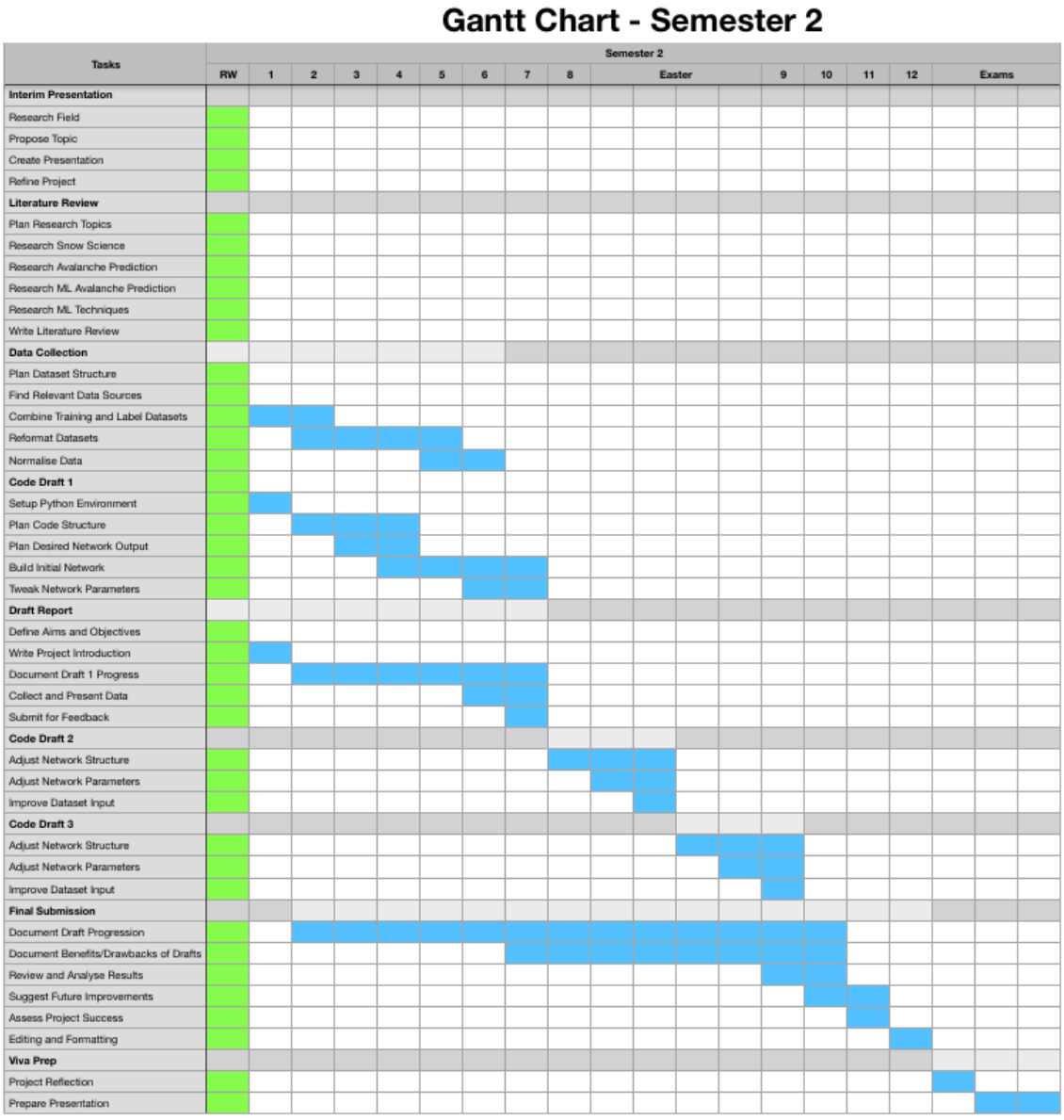

Figure 35 - Gantt chart for Semester 2

6.0 APPENDIX

[1] M. Gassner and B. Brabec, "Nearest neighbour models for local and regional avalanche forecasting", *Natural Hazards and Earth System Science*, vol. 2, no. 34, pp. 247-253, 2002. Available: https://hal.archives-ouvertes.fr/hal-00298992/document

[2] "Dream Job Avalanche Forecaster - An interview with CNFAIC Forecaster", *Off-Piste Magazine*, 2018. [Online]. Available: https://offpistemag.com/dream-job-avalanche-forecaster/. [Accessed: 19- Nov- 2018].

[3] B. Hartline, "Snow Physics and Avalanche Prediction", *Science*, *vol. 203, no. 4378, pp. 346-348,* American Association for the Advancement of Science*,* 1979. Available: http://science.sciencemag.org/content/sci/203/4378/346.full.pdf

[4] S. Bakkehoi, “Snow avalanche prediction using a probabilistic method”, *Avalanche Formation, Movement and Effects, IAHS Publ. no.162 ,1986. Available:* http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.124.9537&rep=rep1&type=pdf

[5] "How is snow formed? | PurePowder", *PurePowder*, 2019. [Online]. Available: https://www.purepowder.com/blog/how-is-snow-formed/. [Accessed: 21- Jan- 2019].

[6] “Snow Science | National Snow & Ice Data Center", *Nsidc.org*, 2019. [Online]. Available: https://nsidc.org/cryosphere/snow/science [Accessed: 21- Jan- 2019].

[7] "Snowpack", *Slf.ch*, 2019. [Online]. Available: https://www.slf.ch/en/snow/snowpack.html. [Accessed: 23- Jan- 2019].

[8] E. Knoff, “Know your angles”, *Gallatin National Forest Avalanche Center*, 2013. [Online]. Available: https://www.mtavalanche.com/sites/default/files/MSA_Slope_Angle_Final.pdf. [Accessed: 23- Jan- 2019].

[9] ”Avalanche risk 4 in Verbier: Safety & Ski Tips", *Freeride Verbier*, 2015. [Online]. Available: http://freeride-verbier.com/blog/avalanche-risk-4-verbier-safety-ski-tips/. [Accessed: 23- Jan- 2019].

[10] "Avalanche Character", *American Avalanche Association / National Avalanche Center*, [Online]. Available: https://avalanche.org/avalanche-encyclopedia/avalanche-character/. [Accessed: 23- Jan- 2019].

[11] "Forecasting and Control", *Global Mountain Solutions Inc.*, [Online]. Available: http://www.mountain-solutions.net/avalanche-services/f/. [Accessed: 06- Feb- 2019].

[12] “Danger Scale", *American Avalanche Association / National Avalanche Center*, [Online]. Available: https://avalanche.org/avalanche-encyclopedia/danger-scale/. [Accessed: 23- Jan- 2019].

[13] "Avalanches, Part 2: Snow Tests", *Recreational Equipment Inc*. [Online]. Available: https://www.rei.com/learn/expert-advice/avalanche-snow-tests.html. [Accessed: 24- Jan- 2019].

[14] “Interpreting Snow Profiles”, *Scottish Avalanche Information Service, SAIS MD 2014,* [Online]. Available: https://www.sais.gov.uk/wp-content/uploads/2014/11/interpreting_snow_profiles.pdf

[15] "KBYG - Know Before You Go Avalanche Safety", *Kbyg.org*, 2019. [Online]. Available: https://kbyg.org/. [Accessed: 21- Jan- 2019].

[16] M. Gassner, H. Etter, K. Birkeland and T. Leonard “NXD2000 - An improved avalanche forecasting program based on the nearest neighbor method”, *Proceedings of the 2000 International Snow Science Workshop (*2000). [Online] Available: http://arc.lib.montana.edu/snow-science/objects/issw-2000-052-059.pdf

[17] K. Kristensen and C. Larsson, “An Avalanche Forecasting Program Based on a Modified Nearest neighbour Method”, *Proceedings of the 1994 International Snow Science Workshop, Snowbird, Utah, USA.* [Online] Available: http://arc.lib.montana.edu/snow-science/objects/issw-1994-022-030.pdf

[18] B. Brabec, R. Meister, N. Raderschall, U. Stockli, A. Stoffel and T. Stucki, “RAIFoS: Regional Avalanche Information and Forecasting System”, *Swiss Federal Institute for Snow and Avalanche Research, Davos Dort, Switzerland (2000).* [Online] Available: http://arc.lib.montana.edu/snow-science/objects/issw-2000-060-065.pdf

[19] R. Dobbins and R. Eberhart, “Neural Network PC Tools”, *Saint Louis: Academic Press Inc., 2014*. [Online] Available: https://books.google.co.uk/books?id=W0OjBQAAQBAJ&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

[20] D. Livingstone, D. Manallack and I. Tetko, “Data modelling with neural networks: Advantages and limitations”. *Journal of Computer-Aided Molecular Design, 11 (1997) 135–142.* [Online] Available: https://link.springer.com/content/pdf/10.1023%2FA%3A1008074223811.pdf

[21] A. Lagandula, “Perceptron Learning Algorithm: A Graphical Explanation Of Why It Works”, *Towards Data Science (2018).* [Online]. Available: https://towardsdatascience.com/perceptron-learning-algorithm-d5db0deab975

[22] A. Chris, “From Perceptron to Deep Neural Nets”, *Becoming Human: Artificial Intelligence Magazine*, [Online], Available: https://becominghuman.ai/from-perceptron-to-deep-neural-nets-504b8ff616e

[23] M. Nielsen, “How the backpropagation algorithm works” in “Neural Networks and Deep Learning”, Determination Press (2015). [Online] Available: http://neuralnetworksanddeeplearning.com/chap2.html

[24] S. Suryansh, “Gradient Descent: All You Need to Know”, *Hacker Noon.* [Online] Available: https://hackernoon.com/gradient-descent-aynk-7cbe95a778da

[25] M. Zaytar and C. Amrani, “Sequence to Sequence Weather Forecasting with Long Short-Term Memory Recurrent Neural Networks”, *International Journal of Computer Applications, vol. 143, 2016.* [Online] Available: https://pdfs.semanticscholar.org/f9ae/308836dc0f96325671be6d75c38a97f42a8b.pdf

[26] S. Hochreiter and J. Schmidhuber, “Long short-term memory.”, *Neural computation*, vol. 9, 1997. [Online] Available: https://www.bioinf.jku.at/publications/older/2604.pdf

[27] T. Chow and C. Leung, “Neural Network based Short-Term Load Forecasting Using Weather Compensation”, *IEEE Transactions on Power Systems, vol. 11, 1996*. [Online] Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=544636

[28] P. Srivastava, “Essentials of Deep Learning : Introduction to Long Short Term Memory”, *Analytics Vidhya, 2017*. [Online] Available: https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/

[29] C. Olah, “Understanding LSTM Networks”, *Colah’s Blog, 2015* [Online]. Available: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[30] O. Buser, M. Butler and W. Good, “Avalanche forecast by the nearest neighbour method” in “Avalanche Formation, Movement and Effects”, *Federal Institute for Snow and Avalanche Research*, 1987. [Online] Available: http://hydrologie.org/redbooks/a162/iahs_162_0557.pdf

[31] Climate Data, “Local Climatological Data”, National Centres for Environmental Information, 2019, [Online]. Available: https://www.ncdc.noaa.gov/cdo-web/datatools/lcd

[32] Avalanche Details, “Avalanche CSV Export”, Utah Avalanche Centre, 2019, [Online]. Available: https://utahavalanchecenter.org/avalanches/details/csv

[33] J. Brownlee, “Loss and Loss Functions for Training Deep Learning Neural Networks”, *Better Deep Learning, 2019*. [Online]. Available: https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/ Precision

[34] S. Saxena, “Precision vs Recall“ *Towards Data Science, 2018.* [Online]. Available: https://towardsdatascience.com/precision-vs-recall-386cf9f89488

[35] S. Narkhede, “Understanding AUC - ROC Curve”, *Towards Data Science, 2018* [Online]. Available: https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5

[36] J. Brownlee, “How to Choose Loss Functions When Training Deep Learning Neural Networks”, *Better Deep Learning, 2019.* [Online]. Available: https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/

[37] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, *Journal of Machine Learning Research, University of Toronto, 2014.* [Online]. Available: http://www.jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf?utm_content=buffer79b43&utm_medium=social&utm_source=twitter.com&utm_campaign=buffer

[38] S. Sharma, “Activation Functions in Neural Networks”, *Towards Data Science, 2017*. [Online]. Available: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

[39] Elite Data Science, “Overfitting in Machine Learning: What It Is and How to Prevent It”, EliteDataScience.com, 2017 [Online]. Available: https://elitedatascience.com/overfitting-in-machine-learning#how-to-prevent

[40] A. Ng, “Preventing" overfitting" of cross-validation data”, Carnegie Mellon University, 1997. [Online] Available: https://ai.stanford.edu/~ang/papers/cv-final.pdf

[41] Utah Avalanche Centre, ”How we generate an advisory”, *Utahavalanchecenter.org*, 2019. [Online]. Available: https://utahavalanchecenter.org/forecast/how-we-generate

[42] J. Brownlee, “Gentle Introduction to the Adam Optimization Algorithm for Deep Learning“, *Better Deep Learning, 2017,* [Online]. Available: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

[43] Keras, “Docs - Optimizers”, Keras Documentation, 2019. [Online], Available: https://keras.io/optimizers/

[44] F. Techel, F. Jarry, G. Kronthaler, S. Mitterer, P. Nairz, M. Pavšek, M. Valt and G. Darms, “Avalanche fatalities in the European Alps: long-term trends and statistics”, WSL Institute for Snow and Avalanche Research SLF, 2016. [Online], Available: https://www.researchgate.net/publication/304562895_Avalanche_fatalities_in_the_European_Alps_Long-term_trends_and_statistics/download

7.0 DEFINITIONS

Snow Pack	A mass of lying snow that is compressed and hardened by its own weight.
Snowflakes	Single ice crystals or clusters of ice crystals that fall from a cloud
Hoarfrost / Surfacehoar	Deposited ice crystals which are found on the snow surface.
Graupel	Round ice crystals between 2 and 5mm in diameter formed by snowflakes which fall through supercooled cloud droplets
Névé	Young, granular snow that has been partially melted, refrozen and compacted
Powder Snow	Dry new snow, which is composed of loose, fresh ice crystals
Cornice	An overhanging accumulation of ice and wind-blown snow, characteristically found on the edge of a ridge or cliff face
Crust	A hard snow surface lying upon a softer layer, formed by sun, rain, or wind
Elaborate Variable	A calculated variable rather than raw data
Preday	Previous daily data readings which are relied upon in order to make predictions.
Perceptron	A node within an ANN which mimics the behaviour of a neuron.
Training Set	The set of data used to train a neural network and find trends in data
Testing Set	The set of data used to test the success of a neural network and the trends which it has been trained on
Sigmoid	A compressive function which maps numbers between 0 and 1.
Tanh( )	A compressive function which maps numbers between -1 and 1.
ReLU	A function which maps any negative values to 0 and leaves all positive variables unchanged
Baseline Accuracy	The accuracy of predictions which can be achieved by always predicting the largest class.

8.0 ACRONYMS

AI	Artificial Intelligence
ANN	Artificial Neural Network
RNN	Recurrent Neural network
LSTM	Long Short Term Memory
KBYG	Know Before You Go - avalanche awareness program
KNN	K-Nearest Neighbour
ML	Machine Learning
MLP	Multi-Layer Perceptron
SLF	Suisse Lawinenforschung - Swiss Institute for Avalanche Research
UAC	Utah Avalanche Centre
NCEI	National Centres for Environmental Information