Optimising a Casper Neural Network Using a Genetic Algorithm to Improve Predictions of Music Genre with Brain Activity

Chamith Edirisinghe

Research School of Computer Science
The Australian National University
Canberra, Australia
u6423750@anu.edu.au

Abstract. Using a subset of data collected from a study at the Research School of Computer Science at ANU, a Casper neural network optimised using a Genetic Algorithm was used to predict the music genre a certain subject was listening to (classical, instrumental or pop). The Casper network is a modification of a Cascade Correlation network, where neurons are added as the model trains if the training progress decreases below a specific threshold and dynamically updates learning rates of certain connections. A genetic algorithm was then employed to optimize the hyperparameters for the model. The method was found to produce a model that did not efficiently predict the correct genre of music, managing to achieve 39% testing accuracy and 32% testing accuracy with a higher mutation rate but is limited by the utilised hardware capability. This does seem to be an improvement of previous classification attempts on the same data due to better generalisation, but it still indicates a problem with the subset of data which is not appropriate for use of reliable prediction of the chosen classes in its isolated form.

Keywords: Neural Network, Casper, EEG, music, music genre, brain activity, brain waves, Genetic Algorithms, hyper-parameters

1 Introduction

It is no secret that music can have a significant impact on your brain activity and function. With its ability to have such an influence your emotions and state of being, music is a powerful medium and has been proven to assist with the recovery of movement in people affected by neurological disorders such as Cerebral Palsy (Thaut, 2005). Using technology such as neural networks and genetic algorithms, systems could theoretically be designed to predict what a person is being stimulated by to better understand how music affects us and to potentially exploit these affects for the use of areas such as disease treatment. By collecting the data of these brain waves and looking for patterns through training a neural network which is what this paper is trying to achieve, it can allow us to get a better understanding on what stimuli correspond to what impacts on our brain and ultimately our entire body.

With the advancements of neural networks, more efficient ways of prediction have been discovered which can be utilized when trying to analyse brain waves due to the vast amount of data that can be emitted. Efficiently identifying features that contribute to reducing loss and generalise well is crucial and will only get better as our understanding of neural networks when to apply certain architectures improve. With the evolution of neural networks however, it comes with added complexity and many machine learning models require the initialisation of many hyper-parameters before you even begin to train your model. Knowing which combination of hyper-parameters result in optimal testing accuracy is crucial in developing an effective model that will be able to predict/classify reliably. In this paper, we explore how Genetic Algorithms can be used to tune these parameters to come up with an improved Casper Neural Network.

2 Background and Method

There were many steps in the method to create a functioning model that uses brain wave data. First, the source data was identified, prepared and a Casper Neural Network was implemented. A genetic algorithm was then implemented to optimise the parameters of the implemented neural network and the most optimal parameters were used to gather the model's testing accuracy from the testing set. The inspiration for using a Genetic Algorithm to optimise hyper-parameters in the Casper Neural Network came from the fact that there are a lot of the hyper-parameters associated with the Casper Neural Network (especially to do with the weight learning rates) and it is not exactly clear what the optimal set up of these parameters are in relation to the dataset being used for this paper or any dataset for that matter. A Genetic Algorithm can provide a simple way to set all these hyper-parameters that will result in a solution that should be better and more efficient than simply trying several settings and seeing which one's work.

2.1 Data Analysis

In a paper published by the ANU Research School of Computer Science, they discovered that the brain activity patterns that arise when you listen to certain genres of music can be distinguished, and models can be created to effectively predict which genre a person is listening to. This is based on a set of features collected from an electroencephalogram (EEG), which is a test that records brain wave patterns using electrodes and this case an Emotic EPOC headset (Rahman et al,

2020). In this paper, a subset of the features captured by the EEG, namely data from the F7 channel, was used to train the Neural Network model which attempts to predict the music genre the subject was listening to during the tests. The F7 channel itself is data collected from the frontal lobe of the brain, produces 25 different features (linear and non-linear) and is classified into 3 classes (music genres). This data was taken from 24 different subjects with 576 entries in total with all classes being assigned an equal number of entries, meaning the dataset is balanced.

2.2 Data Processing

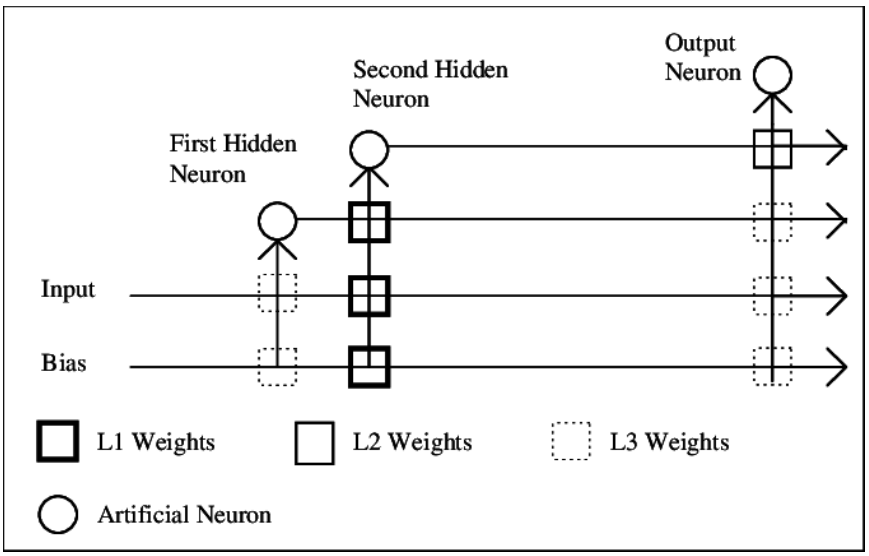
In order to minimize the network's bias towards certain features (and as a result reduce overfitting), the data was first min-max normalized to values from -1 to 1. This is because certain features (such as var_F7) had values that reached 200+ while others (such as mean_first_diff_F7) never exceeded 10 or even less. This way all features would be considered equally.

The data was then randomly shuffled and then split up into a training, validation and testing sets with their sizes being 60%, 20% and 20% of the dataset respectively. The purpose of the validation set is to give a method of testing different hyperparameters on the neural network that is trained on the training set. This should result in a model that generalises better and should ultimately result in higher testing accuracy.

2.3 Casper Neural Network Architecture & Training

The Casper neural network is a modification on the Cascade Correlation (CasCor) neural network. Similar to a CasCor network, Casper does not initially contain any hidden layers and instead adds single neurons throughout the training process. Casper differs from Cascor due to the way it assigns the learning rates of certain weights as shown in Figure 1. Rather than freezing specific weight learning rates which is accomplished in CasCor (Fahlman and Lebiere, 1990), certain values (L1, L2, L3) are used instead.

Figure 1: Diagram showing which weights correspond to which Learning Rates in a Casper neural network (Treadgold & Gedeon, 1997)



The architecture involves first connecting the input neurons straight into the output without any hidden layers. The network is then trained until the loss function (in this case a modification of RPROP) does not improve the loss by 1% in a certain time period. If this happens, a neuron is added to the network which is connected to the input and output. The weights from the input to this neuron correspond to the learning rate L1, the weights from the neuron to the output correspond to L2 and all remaining weights correspond to L3. When another neuron is added, it is connected to the input, output and takes previous neurons as input. All input weights to this neuron are then changed to L1, outputs to L2 and every other weight to L3. Typically, L3 < L2 << L1, as this means that the weights relating to the newly added neuron (L1 and to a lesser extent L2) can learn faster to reduce the remaining network loss. This is repeated for subsequently added neurons until the model finishes training. The time period the network has for improving the loss before decided whether to add a neuron is defined as 15 + P * N where N is the number of neurons added and P is a hyperparameter.

In the actual implementation of the architecture, RMSprop optimizer was used from the python library PyTorch and the learning rates of certain weights were manually set during each addition of a new neuron according to the values of L1, L2, L3. These hyper-parameters, along with the batch size of the input, P and the number of epochs were experimented with set values and a genetic algorithm was also employed to attempt to optimise these parameters.

2.4 Genetic Algorithm

In order to select the hyperparameters for the Casper, a Genetic Algorithm was employed. The GA works by initialising a certain number of individuals. These individuals are made up of genes that represent random values of the hyperparameters of the Casper model, which include the batch size of the input, P and the number of epochs to run. The bounds of these hyperparameters that each individual must conform to is shown in Figure 2. Each individual then trains the Casper NN with their defined genes on the training set in order to build a model. The validation loss produced by running a forward pass of the trained model with the validation set is then calculated. This process of training the model and then calculating the validation loss is done 3 times and the averaged validation loss is used as the fitness function to determine if the individual will make it through to the next generation. The average validation loss is used to come to a more reliable result that should generalise better.

To determine which individuals make it through, a tournament selection was employed, which randomly picks 3 individuals and selects the best one. This is then repeated until the population of the next generation is reached. The initial probability for an individual to mutate a specific gene to a random value was set to 0.05 and to perform a one-point crossover was 0.8.

For the entire process, the software was run with an AMD Ryzen 5 1500X Quad-Core Processor with 3.50 GHz, 8.00 GB of RAM and Microsoft Windows 10 Home 64-bit operating system within a Conda Python environment in PyCharm.

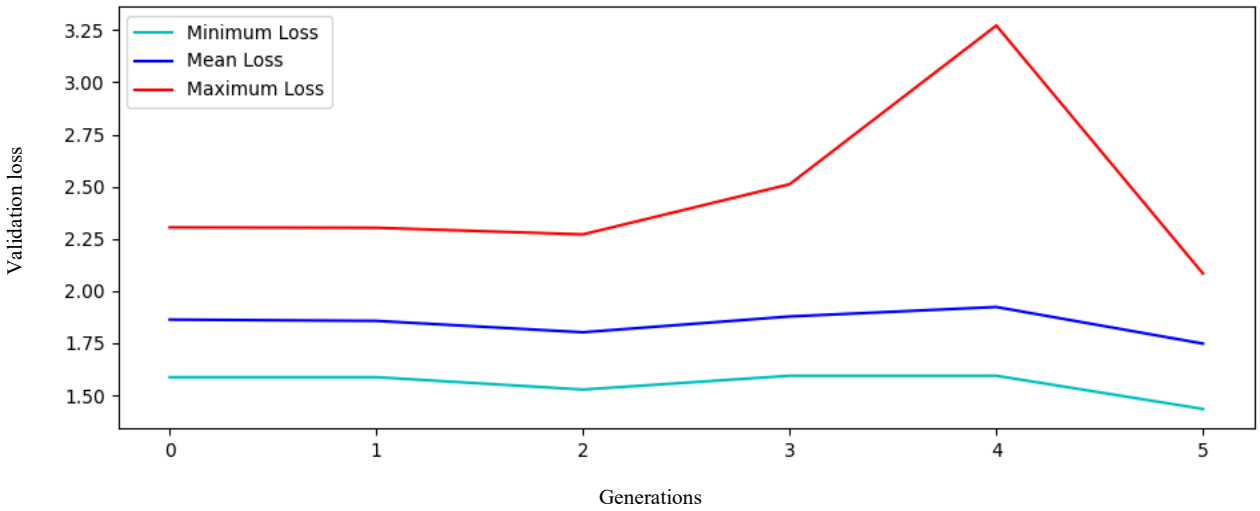
Figure 2. Hyper-parameters of Casper and their possible values within the Genetic Algorithm. L1->L2 multiplier refers to the value needed to multiply with L1 to get L2. This is the same concept with L2->L3 multiplier. This way the Casper model can maintain the property of $L3 < L2 << L1$.

Hyper-parameter	Min	Max
Batch size	1	50
P	1	50
Epochs	100	1000
L1 weight	0.05	0.5
L1->L2 multiplier	0.01	0.08
L2->L3 multiplier	0.1	0.8

Once the GA was completed, the hyper-parameters that resulted in the least validation loss were used to retrain a Casper model which was then used for the test data set to give the final test accuracy. Similar with how the GA individuals averaged the validation loss over 3 trained models, the same was done for the testing accuracy with 3 Casper models being trained with the optimal hyper-parameters, averaging the testing accuracy of all of them at the end.

3 Results

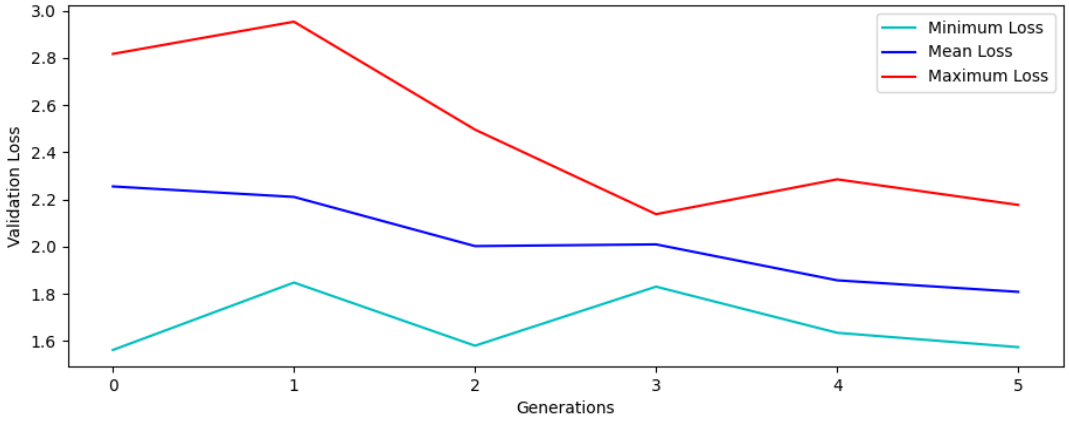
Figure 3. Validation loss across generations of Genetic Algorithm with a crossover probability of 0.8, mutation probability of 0.05, population of 10 and 5 generations.



Even with a genetic algorithm attempting to optimise the parameters of the Casper model, it was not able to minimise the validation loss by a significant margin after the first generation as seen by figure 3. Although there was a slight improvement in generation 5, it does not seem like an improvement that was worth the computation time. With only a population of 10 and using 5 generations, the total computation time to run the genetic algorithm was 31 minutes. This is due to techniques such as averaging the model over 3 trainings and how each individual must train in a sequential order. This means that a population of 10 and 5 generations is equivalent to training the model 150 times (population * generations * averaging runs) one after the after resulting in an inefficient process. This meant that the bounds of hyperparameters such as the batch size and number of epochs had to be limited in order to ensure that the GA terminated in reasonable time.

As the GA with the initial hyper-parameter settings did not seem to improve loss very much, another GA was tested using an increased mutation probability that encouraged randomness (mutations) more in the hopes that it would discover a good set of hyper-parameters for Casper.

Figure 4. Genetic Algorithm run with a crossover probability of 0.8, mutation probability of 0.2, population of 10 and 5 generations.

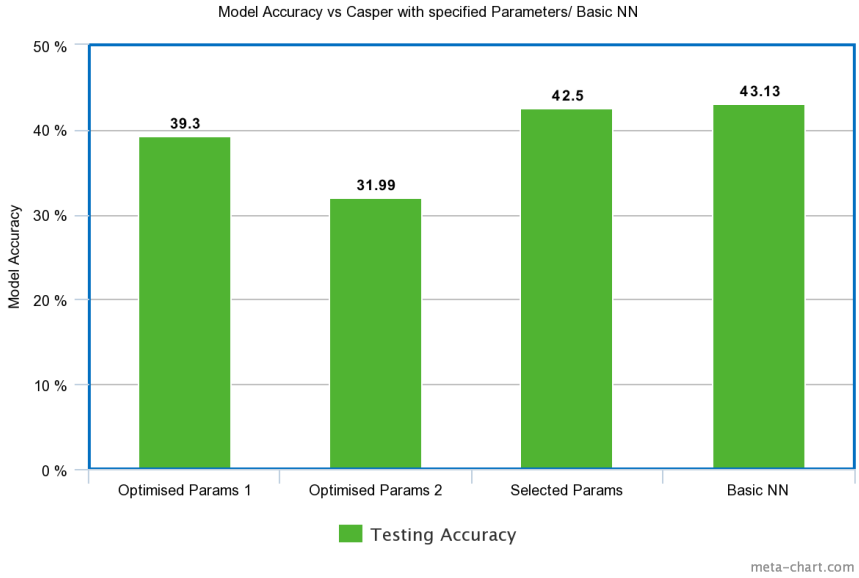


As shown in figure 4, when increasing the mutation probability, the minimum and mean loss becomes more volatile as the model is prioritising exploring rather than exploiting individuals that are known to work. The parameters of the best individual from both GA's can be observed in Figure 5. The individuals of both GA's seem to favour a high number of epochs which agrees with findings in [1]

Figure 5. Parameters of best individual from both GA's.

Hyper-parameter	**Optimised Params 1**	**Optimised Params 2**
Batch size	46	25
P	5	46
Epochs	725	816
L1 weight	0.21	0.34
L1->L2 multiplier	0.013	0.022
L2->L3 multiplier	0.323	0.165

Figure 6. Testing accuracy for Casper models (Optimised Params 1, Optimised Params 2 and Selected Params) and basic neural network. Data for the basic neural network and Selected Params were taken from [1]. Selected params refers to the parameters: batch size = 15, P = 5 and epochs = 1000 and L1, L2 and L3 learning rates of 0.2, 0.005, 0.001 respectively. Basic NN involved a hidden layer with a size of 30 and an RMSprop optimiser with a learning rate of 0.01



When comparing the optimised parameters to the selected parameter Casper model and the basic neural network from [1] in figure 6, it does show worse performance. This however could be due to how the models from [1] were trained without a validation set. Thus, when selecting parameters for the neural networks, they were based on performance on the test set, invalidating it and the models would likely not generalise to an extended dataset with additional subjects.

Optimised Params 1 does perform significantly better than Optimised Params 2, which happens to be worse than a random guess for classification. This may be due to how the mutation probability was too large, increasing the randomness of the individuals in the GA and making the model closer to a random guess. The fact that only a population of 10 and generation of 5 for the GA was feasible for the utilised hardware may have also impacted the results and may have limited the chance for the GA to find an individual that contained hyper-parameters that resulted in a higher testing accuracy.

4 Conclusion

When utilising a Genetic Algorithm to optimise for parameters of a Casper Neural Network, the model gives results similar to but slightly worse than a basic neural network and a Casper network with manually selected parameters. The results gained from the GA however should have better generalisation due to their use of a validation set and would most likely fit better into an extended dataset of the F7 channel part of the EEG. A big downside to this method however is that training Casper models within the GA does take a considerable amount of computation time with the current hardware.

To improve results, parallelisation and the use of GPU servers may aid in allowing the GA to reach its full potential as individuals could train in parrel to reduce computation time significantly (Kavid A. et al, 2018) (even the averaging runs within an individual can be done in parallel). Having greater hardware capability would also allow the extension of hyper-parameter bounds such as batch size and epochs as the rest of the process would take less time and would also make training methods such as K-fold cross validation practical.

Ultimately however, the overall testing accuracy for all tests done so far in this paper and [1] indicate that the data being used from the F7 channel of the EEG is insufficient to train a model to reliably predict music genre listened to and it is unlikely that these methods would boost testing accuracy into values that exceed 80%.

With both the Casper and basic neural networks not working effectively despite a basic neural network working on the superset of data, it is clear that the problem lies within the data itself and that either the data from the F7 channel of the EEG does not correlate very well to music genre listened to in isolation or that a neural network in general is not appropriate for use of reliable classification on this dataset.

References

1. Edirisinghe, Chamith. (2021). Predicting Music Genre with Brain Activity using a Casper Neural Network.

2. Rahman, Jessica & Gedeon, Tom & Caldwell, Sabrina & Jones, Richard. (2020). Brain Melody Informatics: Analysing Effects of Music on Brainwave Patterns. 10.1109/IJCNN48605.2020.9207392.

3. Treadgold, N.K., and Gedeon, T.D. (1997) A Cascade Network Algorithm Employing Progressive RPROP,.. 733-742

4. Kayid, Amr & Khaled, Yasmeen & Elmahdy, Mohamed. (2018). Performance of CPUs/GPUs for Deep Learning workloads. 10.13140/RG.2.2.22603.54563.

5. Fahlman, S.E., and Lebiere, C. (1990) The cascade-correlation learning architecture. In Advances in Neural Information Processing II, Touretzky, Ed. San Mateo, CA: Morgan Kauffman, 1990, pp. 524-532

6. Thaut, M. H. (2005) The future of music in therapy and medicine. *Annals of the New York Academy of Science*, 1060, 303–308