Some optimization techniques to improve the neural network performance

Boyang Zhang

Research school of Computer Science
The Australian National University
Canberra, Australia
u7163241@anu.edu.au

Abstract. Depression is a common phenomenon in the current era. It would be great if we can detect them easily. At emotional level, it can be sensed by others. Meanwhile, depression has different levels, and in [1], researchers set neural networks to predict the depression levels based on physiological signals. They finally achieved 92% accuracy. However, we can still optimize the network by pruning. In [2], researchers focused on the distinctiveness of hidden neurons and removed neurons which are functionally similar and opposite. Results showed that this method reduces the computational consumption of the neural network to a certain extent. In this paper, to predict the depression level by observers' signals, we set a baseline model based on a simple feed-forward network and use some optimization techniques like dropout and early stopping to improve their accuracy. We also use genetic algorithm (GA) [8] to select the most useful features. Then we use pruning to shrink the model size. We reduce about 4 neurons in average and our running time of our model reduce about 5 seconds. So, that means, we find that pruning can actually reduce the computational cost without losing too much performance.

Keywords: Depression detection, Neural networks, Distinctiveness pruning, Dropout, Early stopping, Genetic algorithm.

1 Introduction

Depression is a very serious mental illness, which is usually accompanied by psychological stress and burden. People with depression usually lose confidence in life and are not interested in new things [3]. We need to be very cautious when dealing with depression, because this is not just a bad mood—severe depression can cause serious mental illness and even suicide. Therefore, it is very important to find depression as early as possible, which is conducive to the diagnosis and treatment of psychologists.

1.1 Background

Our commonly used diagnostic methods are generally based on self-reported questionnaires such as the Beck Depression Index (BDI) [5] or clinician assisted interview style assessments such as the Hamilton Rating Scale for Depression (HAMD) [6]. These methods are inaccurate and error-prone. We know that when we diagnose a patient, the truthfulness and accuracy of the questionnaire filled out by the patient will greatly affect the accuracy of the depression level classification. A good example is that for some severely depressed patients, they are depressed and dislike showing true thoughts to the outside world. The information they fill in is not their true thoughts. What's more, manual judgment is very labor-intensive and time-consuming. Therefore, some objective diagnostic methods are urgently needed.

1.2 Motivation and Brief introduction

In previous work like [1], researchers proposed the use of neural networks to help people classify depression. They used a feed-forward network to predict the level of depression based on the observers' signal. Many features were used in their neural networks to solve this classification problem. They actually got an accurate model after many epochs training. However, they used too many neurons and features, which made training time longer and slowed down the entire prediction process. Pruning according to distinctiveness [2] is a good method. In the hidden layer, redundant neurons are removed, and neurons with important functions are retained. The reduction of neurons means a decrease in the computational cost of entire networks. In this paper, we remove neurons which are functionally redundant and compare its accuracy with the previous model. What's more, we have so many features in our dataset and we need select the most useful features. So, we use genetic algorithm here to select features. And, when we try to reproduce model in [1], overfitting is a very serious problem. To be more precise, we use optimization methods such as dropout and early stopping to improve the performance of the model. Finally, we compare multiple groups of models to determine whether these techniques have substantially improved entire neural networks.

1.3 Database

In my model, we use databases collected in [1]. They are collected by different physiological signal sensor on observers' bodies, including Galvanic Skin Response (GSR), Pupillary Dilation (PD) and Skin Temperature (ST). Each database contains 192 rows. All data is obtained from 12 observers. Gender of these participants are evenly distributed. Their age ranges from 18 to 27 years old. The average age is 21.1 years old, with standard deviation 2.8. The data of each observer includes the data of its watching videos from four different depression levels. Videos from 4 different depression levels are evenly distributed, so the database contains 16 pieces of data for each observer. GSR database has 23 features, PD database has 39 features and ST database has 23 features. Features include: *(a) Minimum (b) Maximum (c) Mean (d) Standard Deviation (e) Variance (f) Root Mean Square (g) Means of the absolute values of the first difference (h) Means of the absolute values of the second difference (i) number of particular signals (j)amplitude of particular signals (k) ratio of particular signals* [1]. Some of them are normalized while some aren't. For the depression categories, they have different meanings [1], 0 means no or slight depression (None), 1 means middle-level depression (Mild), 2 means high level depression (Moderate) and 3 means serious level depression (Severe).

2 Method

This section includes some pre-processing methods for the original data in this experiment, the baseline model in this experiment, some optimization techniques and pruning methods. And finally, we introduce the evaluation metrics of these models.

2.1 Pre-processing of data

To make our life easier, we name the column in these 3 databases. Columns containing the participant numbers in these three databases are called "pupil", which makes it easier for me to locate in the subsequent processing. In order to set up the GSR+PD+ST model, we concatenate three databases together to form another database. Otherwise, tables of these four databases have not been changed.

Above are some minor modifications to the externally imported excel files. In this experiment, some of the 85 features are not normalized to 0-1, and some of them are negative. For those columns with negative numbers, we use abs function to change them to the absolute value. While for those columns with data is not between 0-1, we need to utilize the maximum function and the minimum function to make them normalized. The specific formula is:

$$X_i = \frac{X_i - \min(X)}{\max(X) - \min(X)}$$

2.2 Baseline

Our task is to predict the depression level based on input signals, which is a typical classification task. As [1] mentioned, we can use a standard three-layer fully connected feed-forward neural network to achieve this. In this paper, this is our baseline model. The input of this network can be 23, 39 or 85, depending on different databases. Due to space limitations, in this experiment we mainly consider the GSR database in [1] and the optimization effect of pruning on it. Other databases are similar to the results of this database. We have 50 hidden neurons in the hidden layer. The sigmoid function is used as an activation function for hidden-layer neurons. The output layer is composed of four neurons with Softmax activation functions. 4 output neurons 0, 1, 2, 3 respectively correspond to the level of depression. In this case, larger number means more serious depression. In the experiment, the learning rate of our neural network is determined to be 0.01, and the maximum epoch is set to 2000, we use cross-entropy to calculate the loss and output the accuracy every 50 epochs. Adam optimizer is used to update the weights. This feed-forward neural network is achieved by Pytorch.

2.3 Feature selection

At the beginning of the research, we simply test the accuracy of the model, and results are not very satisfactory. We believe it is not because optimization techniques are not used. To a large extent, it may be that the accuracy of our model is reduced due to too many features. If we use all 85 features, these features are not necessarily the most useful features, so some moderate choices may improve the performance of our neural network accuracy. A very natural idea is genetic algorithm (GA) [8]. GA simulates the evolutionary process in biology. In fact, this is an evolutionary algorithm. The GA algorithm sets up crossover and mutation, and uses fitness for selection. The whole process is like the evolution of species in nature. In our experiment, we use GA when we use all features to evaluate the performance.

So, the length of chromosome is 85, the population size is 15, the crossover rate is 0.8, and the mutation rate is 0.0117, which is 1/85. Other parameters are just like the conventional GA settings, the selection is Stochastic uniform selection, the mutation is uniform mutation and crossover is uniform crossover. Here is the detail of GA [10] algorithm.

Algorithm 1. Genetic algorithm

```
Input: maxGenerations, population_size, crossover_rate, mutation_rate
t=0    current generations
p=initializePopulations()
set(mutation rate)
set(crossover rate)
while i<maxGenerations:
    while j<population_size:
        while k<population_size:
            fit=get_fitness
        store.append(fit)
    for parent in pop:
        child=crossover()
        child=mutation()
    fitness.clear()
```

2.4 Optimization techniques

When we try to reproduce feed forward neuron network in [1] with 3 fully connected layers, we find that the results are not ideal. Our model always has some overtraining conditions, such as overfitting. There is no doubt that this affects the accuracy of our model's predictions. We can alleviate the overfitting of the model by adjusting hyperparameters, but this will make the training of our model more time-consuming, because in this case hyperparameters will not be the most perfect. A better way is to use dropout and early stopping.

Dropout is a way to optimize neural networks that was proposed in 2012. As mentioned in [3], when a complex feed-forward neural network is trained on a small data set, it is easy to cause overfitting. In order to prevent overfitting, the performance of the neural network can be improved by preventing the joint action of feature detectors. When we propagate forward, let the activation value of a certain neuron stop working with a certain probability p, which can make the model more generalized, because it will not rely too much on some local features. Figure 1 is the dropout representation.

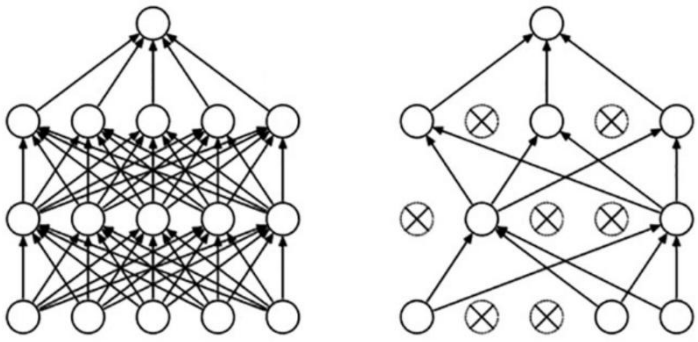

Figure 1. Dropout technique in neural network

On the other hand, in our model, we do not complicate the problem. We use the idea of early stopping method. When the model is training, if the accuracy of the model no longer gains much improvement (more than 1%), we stop training. Because if we continue training, our training accuracy will not be greatly improved. At the same time, our model's performance on the test set will become poor, which is one reason for overfitting.

2.5 Pruning the network

In layman's terms, pruning the network in this paper is to selectively delete some hidden neurons. In fact, dropout also has the same function, but the randomness of dropout makes the accuracy of our neural network unreliable, more details in Result. In [2], the author proposed a method to select pruning objects using the distinctiveness of different hidden neurons. This reduction of the size of the hidden layer is performed by using the distinctiveness angular measure. We take weights from neurons and they are combined in a vector. This important vector describes the function of the

neuron. If two neurons are too similar, then we will remove one of them, because this is a kind of redundancy, which will not only increase our computational consumption, but may also lead our model to overfitting. If functions of two neurons are opposite, then they are redundant as well, and have no meaning other than adding computational burden to our networks. We need to remove this pair of neurons. In [2], the author proposed a good calculation method, which is to compare the angle between two vectors in space. From experience in [2], it is suggested that when the angle between vectors is less than 15 degrees, vectors are considered to be similar. It is worth mentioning that weights of the removed neuron need to be added to the remaining neuron. If the angle between the vectors is more than 165 degrees, we believe that the pair of neurons has opposite functions.

2.6 Evaluation

Due to the different conditions of the participants, we cannot randomly divide these data into training sets and test sets, that is, the data of a participant needs to be gathered in one set. Specifically, we follow the leave one participant out cross validation strategy in [1]. For model evaluation, as in [1], we use four evaluation standards: *precision*, *recall*, *F1 score*, and *accuracy*. *Precision* is defined as the percentage of instances that are correctly predicted for depression level L over the instances that have the depression level L. *Recall* is the proportion of instances that are correctly predicted with depression level L over all instances that have depression level L and *F1 score* takes the harmonic mean of precision and recall. *Accuracy* is defined as the proportion of instances that are correctly predict with depression level L or non-depression level L. In addition, because the database is relatively small, we evaluate the situation of different participants as test sets. In each case, we conduct 10 tests and take the average of the final results. In the end, the mean value of results from all participants as the test set becomes the evaluation standard of our neural network. What's more, we calculate the number of neurons and the change of accuracy after pruning to measure the effect of pruning. We also test the running time of our program. In general, we compare baseline feed-forward neural network, feed-forward neural network with dropout, feed-forward network with early stopping, feed-forward neural network with early stopping and pruning based on 4 different databases and 4 evaluation standards in [1]. Also, as we said, we use genetic algorithm here, we take it into the consideration, we use them to select the feature we use in our neural network and compare the performance. Note that we use the 'all features' model to verify this because this has the best performance.

3 Result and Discussion

Experiments in this paper are conducted on Lenovo Thinkpad E14 with Intel Core i7-10710U CPU@ 1.10 GHz 1.60GHz. Databases in our experiments are not large, so this CPU's performance is considerable. For all databases, optimization techniques and pruning methods can make the performance of neural networks better. There is no doubt that when we use the GSR database, the prediction performance is the best, but the gap of performance with other three databases is not big.

3.1 Effect of dropout and early stopping

Participant	Baseline	Dropout	Early stopping	Early stopping with pruning	Remaining neuron
P02	0.218	0.206	0.269	0.228	47.0
P03	0.281	0.302	0.342	0.335	44.3
P04	0.239	0.243	0.303	0.249	45.3
P06	0.448	0.475	0.499	0.520	47.6
P07	0.261	0.265	0.303	0.303	40.5
P08	0.273	0.270	0.249	0.249	46.0
P09	0.145	0.145	0.169	0.166	47.2
P10	0.385	0.385	0.437	0.458	47.3
P11	0.335	0.363	0.342	0.292	42.7
P12	0.198	0.188	0.187	0.145	47.7
P13	0.393	0.400	0.442	0.442	47.3
P14	0.361	0.361	0.395	0.395	46.2
Average	0.295	0.300	0.327	0.319	45.8

Table 1. Different test sets' accuracy on different networks & average remaining neurons after pruning

The above Table 1 is the accuracy results of different neural networks by using different participants as the test set (using GSR database, others have similar results). The second to fourth columns are the baseline model, the neural network using dropout, the neural network using early stopping, and the Neural network with early stopping and pruning. The bottom row is the average value of each column. For accuracy reasons, all accuracy is expressed in

decimals. In [1], we know that the accuracy of the observer's subjective prediction is 0.27. Our baseline model is higher than this value, but the overall accuracy is not ideal. At the same time, our baseline model is volatile, and its performance in different test sets is not stable. This may be due to the differences in the personal attributes of each observer, such as insignificant pupil changes or relatively stable body temperature. In Figure 2, we can find that our baseline model is overfitted. Dropout and early stopping seem to be some techniques to solve this problem. However, from the average value in the bottom row, we can see that dropout does not significantly improve the baseline model. The accuracy of baseline model is 0.295 while the accuracy of model with dropout is 0.300. It is largely because dropout's random reduction of neurons is not purposeful, and we may remove some important neurons. Due to the different situations of different participants, we need to ensure certain training accuracy, so the epoch must be set at 2000. But in some cases, 2000 epochs will cause overfitting. Here we set an "if" condition to terminate the training early when the training accuracy does not increase. We found that compared to the previous test loss that would be generated in last 1000 epochs, early stopping will control the test loss to a smaller value. Compared to dropout, early stopping has a greater degree of optimization for our neural network. The accuracy of our early stopping neural network is 0.327, which is somewhat higher than our baseline model and dropout model.

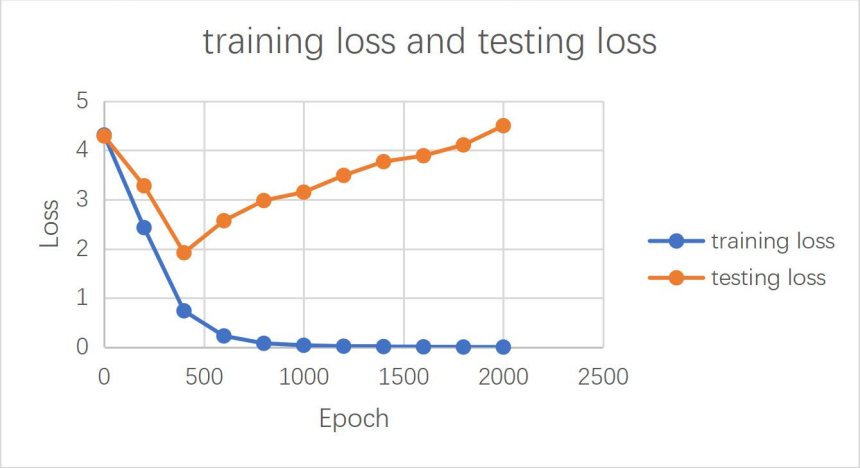

Figure 2. Overfitting problem

3.2 Effect of pruning

In order to avoid removing neurons without brain, we use the vector angle [2] to calculate the distinctiveness of neurons and remove neurons with opposite or similar functions. According to the last 2 columns in Table 1, using pruning does not influence the testing accuracy of network a lot. The accuracy after pruning only reduce 0.008, which is a slightly difference. At the same time, the average number of neurons in our hidden layer is reduced by 4.2, which is not a large number, but this difference is also significant, which means that we usually don't need so many neurons for training. Here, we also record the time it takes to run the program to verify the reduction of computational cost from another perspective, as shown in the following table. It shows that proper pruning can ensure accuracy while ensuring the reduction of our computing consumption.

Participant	All features without pruning	All features with pruning
P02	63	56
P03	53	60
P04	67	57
P06	64	60
P07	54	59
P08	60	51
P09	63	56
P10	61	57
P11	63	58
P12	54	49
P13	63	56
P14	60	48
Average (second)	60.41	55.58

Table 2. Average running time of different participants' testing

3.3 Improvement of performance by using GA

We use GA to select the most useful features. If we use all of the 85 features in 3 datasets to train our model, our accuracy will not be good, so we use GA here. After running our GA code, we choose 81 of the 85 features in our datasets which have the highest fitness. We exclude 4 features overall. Then, we use these remaining features to train our model and get the result.

Participant	All Features	All Features+GA
P02	0.198	0.215
P03	0.289	0.302
P04	0.198	0.245
P06	0.358	0.442
P07	0.241	0.335
P08	0.273	0.366
P09	0.145	0.275
P10	0.336	0.366
P11	0.464	0.425
P12	0.242	0.266
P13	0.366	0.446
P14	0.334	0.392
Average	0.287	0.340

Table 3. The accuracy of testing different participants before and after using GA

3.4 Performance of different databases with different techniques

Using early stopping and pruning, we reduced the computational cost and improved the accuracy of the model. In the previous table, our comparisons are based on the GSR database. Now, our analysis goes further. In Figure 3 and Table 2, we compare the *precision*, *recall*, *F1 score* and *accuracy* based on different databases, we can see that GSR database performs best, but the gap between them is not big. Here we can find an interesting fact GSE+PD+ST database does not give us best prediction, and we can see the great improvement after using GA, that is: all features cannot guarantee the accuracy of our model. This also lets us know that choosing some important features by genetic algorithm is better than blindly using all features.

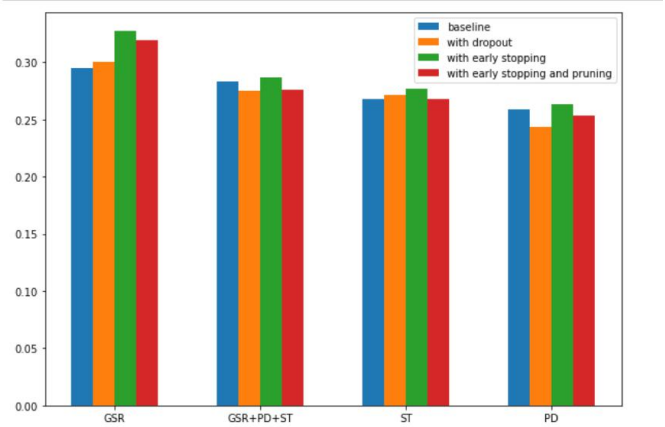

Figure 3. Accuracy of different model using different databases

Depression level	GSR			All Features			ST			PD		
	Precision	*Recall*	*F1 score*	*Precision*	*Recall*	*F1 score*	*Precision*	*Recall*	*F1 score*	*Precision*	*Recall*	*F1 score*
None	0.23	0.35	0.28	0.26	0.28	0.27	0.23	0.31	0.26	0.23	0.31	0.26
Mild	0.19	0.26	0.22	0.21	0.26	0.23	0.19	0.25	0.22	0.26	0.26	0.26
Moderate	0.32	0.35	0.33	0.28	0.36	0.32	0.22	0.33	0.26	0.33	0.31	0.32
Severe	0.40	0.31	0.35	0.33	0.31	0.32	0.35	0.35	0.35	0.19	0.21	0.20
Accuracy	0.327			0.287			0.277			0.263		

Depression level	All Features+GA		
	Precision	*Recall*	*F1 score*
None	0.25	0.34	0.29
Mild	0.25	0.29	0.27
Moderate	0.39	0.41	0.39
Severe	0.43	0.33	0.38
Accuracy	0.340		

Table 4. Performance of neural networks with early stopping based on different databases with GA+All Features

This criteria in [1] of evaluating the model is reasonable, combining the *F1 score* and the overall accuracy, we can clearly see the accuracy of prediction for different levels of depression. We can find that the higher the level of depression, the easier we can predict. At the same time, prediction of lower-level depression has a larger gap between *precision* and *recall*. I think this may be due to the volatility caused by low-level depression, because their characteristics are not obvious and it's hard for observers to judge.

3.5 Discussion

First, for optimization techniques, we usually use dropout and early stopping to optimize our neural network. In [4] and [7], researchers combined the idea of dropout with a variety of neural networks and achieved good results. However, here, our dropout has no effect on our feed-forward neural network, and the accuracy is only improved by 0.5%, which seems to be a measurement error. No matter if I set the value of dropout to 0.2, 0.5 or 0.8, we cannot optimize our network. Of course, there is the instability of human testing, but I think this is due to the randomness of dropout which makes us unable to know whether it will make neurons with important functions removed. Compared to dropout, early stopping is a better method. As shown in Figure 2, we hope to stop training at the inflection point where the test loss is the lowest, so as not to overfit our neural network. Facts have proved that this technique is effective.

Another thing is pruning. We use the method in [2] to convert weights of hidden neurons into a vector, and analyze the distinctiveness by calculating the angle between vectors. Results tell us that the decrease in accuracy caused by pruning is very small, but at the same time, we find that a few neurons are reduced, which proves that pruning is effective and it can reduce our computational consumption. What's more, we find that the average running time reduce by 5 seconds after using pruning on our neural network, it also means that the computational cost is saved by using pruning.

Finally, we compare the performance of different databases. It is easy to find that GSR database has the best performance. It has only 23 features. On the contrary, PD (39 features) and GSR+PD+ST (85 features) are not as good as GSR database, which shows that PD and ST databases will reduce the accuracy of our original GSR model. The accuracy of using all features is 0.287 and the accuracy improve to 0.340 after using genetic algorithm. It shows the great influence of GA. It means that if we can choose some features appropriately, the accuracy of the model will become higher. Of course, the result will be similar if we use GA [9] on other datasets.

However, in [1], the best model achieved an accuracy of 92%, and even the simplest neural network using all features achieved an accuracy of 88%. After I tried my model a few times, I was shocked by this gap. In fact, my baseline model is intended to be a replica of model in [1], but this accuracy gap is surprising and obvious. I think my neural networks are defeated by networks in [1] for two main reasons: 1) Our measuring methods are different. It may be that the difference in measurement causes such a huge difference in accuracy. 2) My model cannot be called the replica of model in [1]. Settings in some parts may not be satisfactory, for example, the normalization setting may be different from that in [1]. But it can still be found that our several optimization methods have produced a certain optimization effect on our model. This shows that these techniques are meaningful.

4 Conclusion and Future work

In this paper, in order to solve the problem of predicting depression levels, we create a three-layer fully-connected neural network. Subsequently, we evaluate the impact of dropout and early stopping on the prediction accuracy, and results show that early stopping is the most suitable one for the optimization of this network. More importantly, we verify the function of pruning by distinctiveness, and results show that this pruning can reduce the number of neurons while ensuring accuracy. It can also reduce the time consumed in the training, thereby reducing computational consumption. In addition, GA used in our model can play an important role, it largely improves the accuracy of the model by selecting the most useful features. Our model only reaches 40% of the accuracy of the model in [1], so there is still room for improvement.

In future work, we can obtain more data from more observers to assist our neural network in training. We only have 12 observers' data here and each observer only have 16 rows of data. If we have more data, other accuracy may improve. At the same time, I find that features in datasets are a bit too redundant. Many features have the same meaning. Although the GA can help us select, I think it is more important to extract some more representative features. We can also use some regularization methods to improve our model. Actually, we think the pruning method can be improved. When dealing with large neural networks, we may need to delete more neurons, which is a test of the performance of the pruning method. Also, we have many evolutionary algorithms up to now. Other EA can also be used here to be compared. But in general, we think we can use more advanced technology to help us classify depression in the future.

References

[1] Xuanying Zhu; Tom Gedeon; Sabrina Caldwell and Richard Jones. "Detecting emotional reactions to videos of depression". In: *IEEE International Conference on Intelligent Robots and Systems* (2019).
[2] T. D. Gedeon. "Indicators of hidden neuron functionality: the weight matrix versus neuron behaviour". In: *Proceedings 1995 Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems, Dunedin, New Zealand, 1995, pp. 26-29, doi: 10.1109/ANNES.1995.499431.*
[3] Hinton, Geoffrey E., et al. "Improving neural networks by preventing co-adaptation of feature detectors." *In: arXiv preprint arXiv:1207.0580 (2012).*
[4] Baldi, Pierre, and Peter J. Sadowski. "Understanding dropout." In: *Advances in neural information processing systems 26 (2013): 2814-2822.*
[5] A. T. Beck, R. A. Steer, and G. K. Brown. "Beck depression inventory-II". In: *San Antonio, vol. 78, no. 2, pp. 490–498, 1996.*
[6] M. Hamilton. "A rating scale for depression". In: *J. Neurol. Neurosurg. Psychiatry, vol. 23, no. 1, p. 56, 1960.*
[7] Wang, Sida, and Christopher Manning. "Fast dropout training." In: *International conference on machine learning. PMLR, 2013.*
[8] Holland, John H. "Genetic algorithms." Scientific American 267.1 (1992): 66-73.
[9] Sivanandam, S. N., and S. N. Deepa. "Genetic algorithms." Introduction to genetic algorithms. Springer, Berlin, Heidelberg, 2008. 15-37.
[10] Davis, Lawrence. "Handbook of genetic algorithms." (1991).