Neural Network Reduction Heuristics in Classification Problems

Haikuan Liu

April 2018

1 Abstract

Artificial neural network(ANN) can now be applied to solve a wide range of problems like pattern recognition, generating synthetic data, etc. However, the existence of redundant features within training data and inappropriate hyper-parameter may jeopardize the training speed or the performance of an ANN. For example, too many hidden neurons may lead to overfitting problem and ANN with insufficient hidden neurons may not well model the complexity of the ground true patterns. In this paper, methods of feature selection and how to tune the hyper-parameters for ANN are discussed and reported. This study used both the rule based statistical methods(compute and evaluate statistical features of an ANN) and machine learning method(directly evaluate the loss on different settings of an ANN) for ANN reduction. Specifically, the statistical reduction methods contains *error sign testing*(EST, details in Section 3.3) and evaluating hidden neurons' *distinctiveness*(details in Section 3.4) and the machine learning method applied *genetic algorithm*(GA, details in Section 3.5) to find best feature and hyper-parameter setting. All methods are applied and compared on a single one hidden layer ANN for classification. The results show that by using the statistical reduction methods, the pruned ANN can increase the training speed by 35.7%(from $0.145s$ to $0.092s$) while maintaining the relatively high accuracy(95.17%) compared with other statistical machine learning methods(91.4% for *support vector machine* and 87.7% for *k nearest neighbor*[1]); by using GA for feature selection, the ANN tends to perform better than other methods(96.36% accuracy). The results show that we could use methods discussed in this paper to greatly improve the training time performance and the prediction performance of an ANN. And the methods could be further applied or extended for other neural network structures.

keywords: neural network reduction, feature selection, hyper-parameter tuning, genetic algorithm

2 Introduction

An assumption in this paper is that all the Neural Network mentioned is a fully-connected feed forward neural network.

In general, an over complex model tends to better fit the training data. However, it usually performs worse on the testing data or generating synthetic data. This problem is known as overfitting in machine learning terminology. In statistics, it is called the balance between bias and variance. [2] shows how to decompose bias and variance from loss function which is to be minimized. A good model should be in a good balance between bias and variance. Some statistical approaches on how to prune an ANN are also discussed in [2] and [3]. Specifically, I will talk about how to implement or adopt some approaches(EST for pruning training data, reducing hidden neurons by computing distinctiveness and genetic algorithm for feature selection and hyper-parameter tuning) in detail to solve classification problems.

Besides we can evaluate the statistical features(e.g., distinctiveness) of an ANN, we could also judge whether an ANN has appropriate hyper-parameter setting or feature set by directly evaluating the loss(e.g., mean squared error) or the measure of its output(e.g., accuracy). An near-optimal approach is to use GA[4]. We could apply GA to a population of different hyper-parameter settings or training data settings and obtain the optimized values for better performance. For example, GA can be adopted for selecting the features of training data to get better result[4]. Other researches use GA to optimize hyper-parameters of classifiers. For example, researchers found near-optimal hyper-parameters for support vector regression in forecasting tourism demand [5]

Since the complexity of ANN has been reduced, its ability of describing complex patterns may shrink. So how much we should reduce the ANN highly depends on the patterns we get. The constraints of network reduction methods are also illustrated in [2]. In addition, reduction methods may not improve the prediction performance but can speed up the training time. So the prediction performance and the time performance of

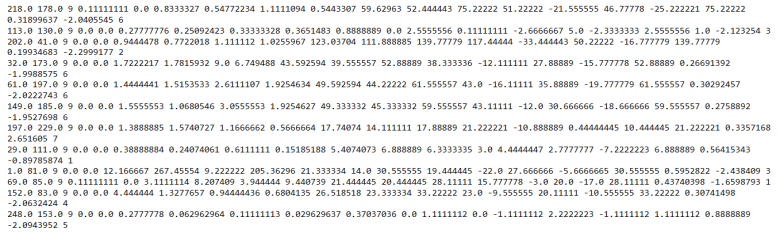


Figure 1: Samples of dataset

the ANN should be both considered. The measurements of the prediction performance may vary with the type of problem we encounter. In [6], the paper discussed different measurements for different kind of problems and their limitations. I choose to use the accuracy and F1-score as my measurement(which will be described in method section) for the classification problems. Besides, [7][8] talked about how cross validation can help us obtain robust model and convincible results. Cross validation is also applied to my experiments.

The rest sections of this paper are organized as follows: I will discuss some considerations when constructing a ANN and two reduction methods for ANN pruning in Section 3. In Section 4 talks about how those methods perform on our actual image prediction problem. Finally, inspirations from experiments and possible future work will be discussed.

3 Materials and Methods

3.1 Data

This study uses the Image Segmentation data set from UCI machine learning repository with 2310 instances, 20 attributes is created by C. Blake, E. Keogh, and C. J. Merz in Vision Group, University of Massachusetts[9]. The instances were drawn randomly from a database of 7 outdoor images. Each instance contains 19 real number attributes('region-centroid-col', 'region-centroid-row', 'region-pixel-count', 'short-line-density-5', 'short-line-density-2', 'vedge-mean', 'vegde-sd', 'hedge-mean', 'hedge-sd', 'intensity-mean', 'rawred-mean', 'rawblue-mean', 'rawgreen-mean', 'exred-mean', 'exblue-mean', 'exgreen-mean', 'value-mean', 'saturatoin-mean', 'hue-mean') of a 3*3 pixels region of a image. The last attribute is a integer attribute indicates the image classes which are brickface, sky, foliage, cement, window, path, grass corresponding to 1, 2, 3, 4, 5, 6, 7 respectively. Sample of the data set is shown by Figure 1 The 19 real number attributes are normalized into zero-mean and unit variance.

3.2 Training process for artificial neural network

The data mentioned in Section 3.1 was processed by 5 fold cross validation(details in Section 3.6) during training and testing. Each fold contains 1848 data points for training and 462 data points for testing.

For the data set, there are 19 features to be learned in each data point and 7 different classes, so the neural network requires 19 input neurons and 7 output neurons. Since the data set has only 19 attributes and relatively simple, we initialize the number of hidden neurons as 8. The cross entropy loss which is commonly used in the classification problems[10] is chosen to evaluate the output of the ANN. The cross entropy can be computed by the equation 1

$$L = -\sum_{c=1}^{M} y_{o,c} log(p_{o,c}) \tag{1}$$

Note that M is the number of classes, $y_{o,c}$ is binary indicator shows whether a data o belongs to class c, $p_{o,c}$ denotes the probability of data o belongs to class c. Because of the multi-class property of the data set, the softmax function is used to normalize the 7 dimensional output vectors to let each element is bounded within [0,1] and the sum of the elements is 1. The softmax function[11] is defined as

$$\sigma(\mathbf{z}) = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad where \quad j = 1, ..., K \tag{2}$$

To update our ANN model in each iteration, proper learning algorithm is required. Rprop algorithm is one of the adaptive and fast algorithm, and it avoids the inherent problem existed in pure gradient descent algorithms[12].

So in this study, the Rprop is chosen to be the weight updating algorithm.

The behaviour of this original unpruned NN is shown by Figure 2. Loss for both training set and test set is decreasing rapidly, then the test set loss starts to increase since epoch 50 because of overfitting, so the number of epoch is set to 50 in order to avoid overfitting. Section 3.3, 3.4 and 3.5 will introduce techniques to prune the initialized ANN discussed in the previous paragraph.

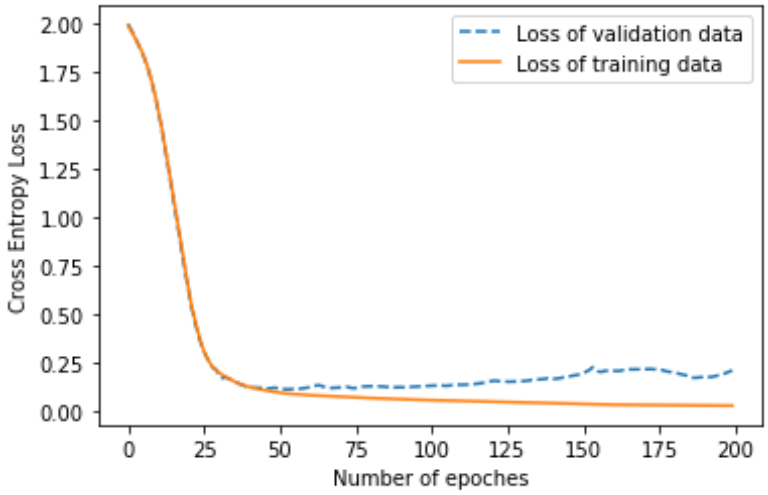


Figure 2: The changes of cross entropy loss of an neural network

3.3 Error Sign Testing(EST)

EST is known as a training set pruning method whose main aim is to remove the noisy training data points. First, during training, we need to find a proper epoch as a threshold that after this epoch training may start to fit the outliers/noisy data[2].Such epoch could be found by inspecting Root-Mean-Squared-Error(RMSE) of loss. I use the same method to detect such an epoch, as shown by Figure 3. We could see that at epoch 50, the value of RMSE starts to oscillate.

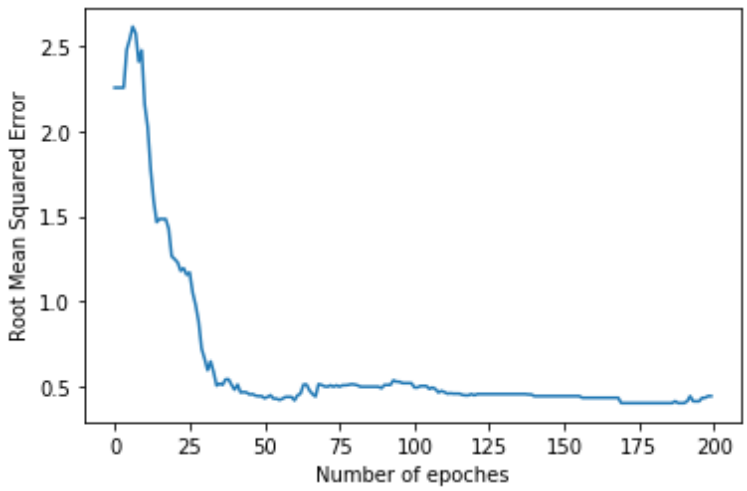


Figure 3: Example of how to choose a proper epoch

To save training time, when RMSE starts to oscillate, we could stop training, record the epoch number K, then apply the pruning method described in the next paragraph and remove redundant patterns. Then retrain the network using the clean dataset.

Within the K epochs, we compute the error vectors formed by differences between desired and predicted vector for each pattern. The desired vector is the target value(0-7) represented by one-hot vector. For example,

$t_3 =< 0,0,1,0,0,0,0 >$ where t_n denotes target value n. The predicted vector is the vector obtained from output layer. The magnitude of the error vector of a good pattern is suppose to decrease in most of the K epochs, see [2]. Let E_n denotes the magnitude of error vector computed at epoch n. The sign of $E_n - E_{n-1}$ is recorded to represent whether the pattern move closer to desired output. Finally, the percentage of how many negative signs shows up within the K epochs is computed to represent the general behaviour of the pattern. Those patterns with low percentage negative error signs are regarded as noisy pattern and removed.

3.4 Distinctiveness

We can reduce the number of hidden neurons by evaluating the distinctiveness between each pair of hidden units. A hidden activation vector formed by activation output of the neuron for each pattern is recorded. This vector shows the functionality of this hidden unit, see [3]. We normalize the vectors to center them at 0.5,...,0.5 so that the angle between two vectors can be scaled to $[0^o, 180^o]$ instead of $[0^o, 90^o]$, then compute the angle distances between each pairs. A pair of vectors with relatively low angle separation(like 15^o) are regarded as similar. It means the corresponding hidden neurons have similar functionality, so one of the neuron is removed and its weights are added to the other neuron or we cut the number of hidden neurons by one and retrain the network. A pair of vectors with relatively high angle separation(like 165^o) are regarded as complementary. The corresponding hidden neurons are less functional than other neurons, so all the two will be removed.

3.5 Genetic algorithm

We can optimize the feature set and hyper-parameter setting for our ANN by directly evaluating the loss. One of such optimization techniques is GA. GA contains bio-inspired operators like crossover, mutation and selection[13]. By applying these operators to the initialized population, we can gradually move towards the optimal solution.

To apply GA to our ANN, the population should be initialized first. How to initialize the population highly depends on the form of the target we want to optimize. In this study, we want to find the optimal subset of features and the optimal number of hidden neurons for the ANN. So a simple binary vector can be used to represent each population.

Specifically, in this study for feature selection, the 19 features of each data point can be represented by a 19 dimensional binary vector where '0' element means the disappearance of the corresponding feature and '1' means the appearance of the corresponding feature. For deciding the number of hidden neurons, similar binary vectors can be used but its elements is used to compute the number of the hidden neurons. For example, 5 dimensional vector $[0,0,1,0,1]$ can be converted to integer 20 by following $2^0 \times 0 + 2^1 \times 0 + 2^2 \times 1 + 2^3 \times 0 + 2^4 \times 1 = 20$ and a 5 dimensional vector can represent maximum 31 this way.

Note that, both feature set and the number of hidden neurons can be represented by one binary vector. This can be achieved by splitting the binary vector into sub-vectors. For example, a 24 dimensional binary vector are selected to find optimal feature set and the number of hidden neurons. There are 19 features and a maximum 31 hidden neurons. The 24 dimensional binary vector can be split into sub-vectors with 19 dimensions and 5 dimensions. The 19 dimensional sub-vector represents feature set. And the 5 dimensional sub-vector represents the number of hidden neurons

By randomly generating such binary vectors 100 times, thus a population with size 100 is generated. Then operators can be applied to the population. Selection is to find the fittest individual which will be operated by crossover. The tournament selection is chosen in this paper because it can make use of the fitness information of each individual and the diversity of chromosomes(features) is also kept by running 'tournaments'[14]. Crossover is to pass the fittest individual's chromosomes(features) to the next generation. The simple two-point crossover is adopted in this paper because there is no theory proves other crossover approaches outperform the two-point method[15]. Mutation is to maintain the diversity of chromosomes(or explore fitter chromosome). Flip bit mutation is used in this paper because the individuals is represented by binary vectors.

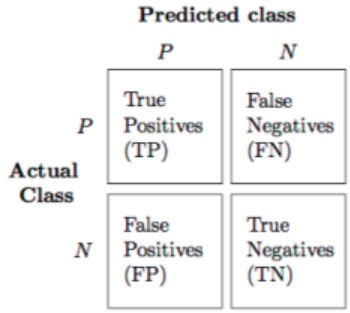
The fitness is chosen as the classification accuracy of our ANN model because our target is to improve the classification accuracy. To obtain a relatively promising result in a short time period, the population size is set to be 20, the generation is set to be 10 and the mutation rate is set to be 0.2 consider the complexity of the data set is not quite high, but these parameters can be further tested and compared in the future.

Toolbox for employing GA in this paper is python package 'Deap'[16]. In this paper, GA is only used to select the best feature set because hyper-parameter selecting requires high dimensional population vector which

consumes lots of time to find the optimal vector. But how to select appropriate hyper-parameter has been discussed in the precious two paragraphs.

3.6 Measurements and Validation

In this paper, I will use accuracy, precision, recall, F1 score as our measurements because they can capture different aspects of classification problem. Figure 4 shows how to construct such a matrix for two class classification problem.



TP is correctly predicted event values

FP is incorrectly predicted event values

TN is correctly predicted no event values

FN is incorrectly predicted no event values

Figure 4: Confusion matrix for two class classification problem

There are 5 ordinal classes to be classified in our study, so the results for each class need to be *macro-averaged* or *micro-averaged* in order to compare results for the 5 classes as one measure. Macro-averaging just takes the average of the precision and recall of the system on different classes. For micro-averaging, we sum up the individual true positives, false positives, and false negatives for different classes and then apply them to get the accuracy, precision and recall. In experiments, we mainly focus on results computed by macro-average method because we want to know how our model performs across all of the classes[7]. The method is described in [7] and follows the following computation:

$$Accuracy = \frac{\sum_{k=1}^{K} TP_k}{N} \tag{3}$$

$$Macro - average \quad of \quad precision = \frac{\sum_{k=1}^{K} P_k}{K} \tag{4}$$

$$Macro - average \quad of \quad recall = \frac{\sum_{k=1}^{K} R_k}{K} \tag{5}$$

where N denotes the number of all predicted values, K denotes the number of classes, P_k is the precision for class k, R_k is the recall for class k.

To make most use of the data, and to obtain convincible results, 5-folds cross validation is used in experiments. That is, the training set is separated into five equal sized subsets. Five iterations are run and in each iteration, we select a different subset(validation set) for testing and use the rest four subsets for training. There are five results derived from testing on the five different validation sets. Those results are also averaged to get the final results. The tools used for cross validation is python scikit-learn package described in [17].

4 Results and Discussion

For pattern reduction, the epoch K highly relies on the learning rate. It is reasonable because with a larger learning rate, the loss will go down faster and the oscillation problem will arise earlier. So I first run 5-fold cross validation to find a proper learning rate, the result is shown by Table 1. The learning rate with the highest accuracy is selected, in this case, 0.01.

Learning Rate	Accuracy/%
0.01	95.97
0.02	95.49
0.03	94.98
0.1	93.43

Table 1: Results for different learning rate

Then by testing the negative error sign percentage, 631 patterns whose negative error sign percentage below 15% are removed.

For hidden units reduction, the activation vectors are computed, the result for the first 8 patterns is shown by Figure 5.

	HiddenUnit0	HiddenUnit1	HiddenUnit2	HiddenUnit3	HiddenUnit4	HiddenUnit5	HiddenUnit6	HiddenUnit7
0	-0.124816	-0.057217	-0.178779	-0.139129	-0.014862	-0.098848	0.113614	-0.300000
1	0.101704	-0.265925	-0.159230	0.040442	0.183178	-0.043033	-0.178997	0.342771
2	-0.261697	-0.035993	0.299630	0.291197	-0.293739	-0.158556	-0.296025	0.082458
3	-0.038147	0.289375	-0.097326	0.042954	0.106021	0.114388	0.099427	-0.100424
4	-0.209005	0.108783	0.104694	-0.108488	0.265945	-0.097518	0.099790	-0.099990
5	0.096601	-0.095861	0.233670	-0.413742	-0.299959	0.043818	0.093823	0.299934
6	-0.267591	-0.442890	-0.239892	0.113771	0.499283	-0.100078	0.315043	0.074187
7	0.166620	0.109320	-0.169685	0.283770	0.071247	0.279131	0.316944	0.177642

Figure 5: Activation vector for the first 8 patterns

Then the angle between each pair of hidden neuron activation vectors is computed, shown by Figure 6.

```
0 1   85.459          2 4   71.4722
0 2   164.24986       2 5   96.45926
0 3   101.722466      2 6   101.165924
0 4   108.09507       2 7   132.9793
0 5   88.259926       3 4   108.26589
0 6   79.33924        3 5   78.97239
0 7   50.939133       3 6   47.01897

1 2   97.01047        3 7   82.82205
1 3   77.808075       4 5   100.24521
1 4   65.99174        4 6   105.95421
1 5   99.93753        4 7   103.22966
1 6   76.627525       5 6   82.200226
1 7   83.57416        5 7   68.694954
2 3   79.44226        6 7   68.33448
```

Figure 6: Angle between each pair of hidden neuron activation vectors

We can see that the pair '0 2' has angle 164^o which are far more larger than the angle of the other pairs, so these two neurons are complementary neurons and both of them are removed. For feature selection by using ge-

netic algorithm, the best individual feature set is represented by vector $[1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1]$, that is, the 4th, 5th and 15th features are eliminated.

The speed performance and complexity between the original NN and the pruned NN is record by Table 2.

	Training Time/s	Training Patterns	Hidden Neurons
NN	0.145	1848	8
Pruned NN	0.092	1217	6

Table 2: Comparison between original NN and pruned NN

We could say that, the training time are reduced significantly after pruning. However, for feature selection by using GA, the process is quite time consuming, so only the training time for EST and distinctiveness methods are reported in this paper.

Then the accuracy, precision, recall and F1 performance is compared by Table 3.

	Accuracy/%	Precision/%	Recall/%	F1/%
NN	95.76	95.77	95.73	95.72
Pruned NN(EST+distinctiveness)	95.17	95.56	93.98	94.59
Pruned NN(GA)	96.36	96.85	96.01	96.43
kNN[1]	87.7	/	/	/
SVM[1]	91.4	/	/	/

Table 3: Comparison between different algorithms, the kNN and SVM result are from [1]

Note that results on the same data set derived from [1] is also recorded. In [1], support vector machine and k nearest neighbor algorithms are used for classifying the images. From Table 3, we could observe that the accuracy of the pruned NN(95.17%) is almost the same as the original NN(95.76%). And compare the NN methods with the statistical machine learning methods, we find that NN methods both lead to a better result. The precision between the two NN methods is pretty close. Only the recall of the pruned NN(93.98%) is much lower than the original NN(95.73%). We can obtain the best result by applying GA for feature selection, all four measures are outperform other statistical or simple feed forward NN methods(96.36% accuracy, 96.85% precision, 96.01% recall and 96.43% F1 score). However, GA requires much longer time to get the fittest feature set.

4.1 Conclusion and Future Work

By removing outliers in the training set, the prediction performance of the model should increase. However, we could tell from Table2 that the pruned NN performs slightly worse than the original one. This may due to I over dropped the patterns and the hidden neurons. So the threshold of reduction(negative error sign percentage or the angle between hidden neuron activation vectors) plays an important role in the reduction process. We should be very careful when determine their values.

Though the predicted performance of the pruned NN is slightly worse than the original one, the speed of pruned NN could increase a lot. The results show that the reduction heuristic can improve the efficiency of a NN.

By applying GA for feature selection, we could get the best accuracy which is 96.36%. The result shows

that the performance of classifier highly depends on the feature set and GA is a good candidate to find the optimal feature set. However, we should take time into account when applying GA to an ANN model.

Although approaches discussed in this paper can efficiently improve the performance of an ANN model, there are also limitations within this study.

This paper only brings up the idea of hyper-parameter selection by using GA, further experimentation should be taken to validate the idea. The hyper-parameter of GA(population size, the number of generation, mutation rate, etc.) should also be tested and selected carefully. And this paper only introduces the heuristics based on experimentation, further theoretical proof needs to be given.

Since we have inspected the statistical methods(EST and distinctiveness) of pruning patterns and selecting the number of hidden neurons in a certain layer, we may develop other statistical methods for evaluating other hyper-parameters(e.g., the number of hidden layers) in the future. As the problems handled by neural network are more and more complex, the neural network tends to be deeper and deeper, so methods of controlling or analyzing deep neural network is required by now or the near future.

References

[1] J. T. Y. Kwok. Moderating the outputs of support vector machine classifiers. *IEEE Transactions on Neural Networks*, 10(5):1018–1031, Sep 1999.

[2] T. D. Gedeon, P. M. Wong, and D. Harris. Balancing bias and variance: Network topology and pattern set reduction techniques. pages 551–558, 1995.

[3] T.D. Gedeon and D. Harris. Network reduction techniques. *Proc. International Conference on Neural Networks Methodologies and Applications*, 2:25–34, 1991.

[4] Jihoon Yang and Vasant Honavar. Feature subset selection using a genetic algorithm. pages 117–136, 1998.

[5] Kuan-Yu Chen and Cheng-Hua Wang. Support vector regression with genetic algorithms in forecasting tourism demand. *Tourism Management*, 28:215 – 226, 2007.

[6] Marketta Hiissa Filip Ginter Shuhua Liu Dorina Marghescu Tapio Pahikkala Barbro Back Helena Karsten Tapio Salakoski Hanna Suominen, Sampo Pyysalo. Performance evaluation measures for text mining. 2.

[7] Hanna Suominen, Tapio Pahikkala, and Tapio Salakoski. Critical points in assessing learning performance via cross-validation. pages 9–22, 2008.

[8] Ron Kohavi. A study of cross validation and bootstrap for accuracy estimation and model selection. pages 1137–1143, 1995.

[9] E. Keogh C. Blake and C. J. Merz. Uci repository of machine learning databases,. 1998.

[10] Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134(1):19–67, Feb 2005.

[11] Wikipedia contributors. Softmax function — Wikipedia, the free encyclopedia, 2018. [Online; accessed 29-May-2018].

[12] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: the rprop algorithm. pages 586–591 vol.1, 1993.

[13] Mitchell Melanie. *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1996.

[14] Brad L. Miller, Brad L. Miller, David E. Goldberg, and David E. Goldberg. Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems*, 9:193–212, 1995.

[15] Raj K. Pant Pravir K. Chawdhry, Rajkumar Roy. London: Springer, 1998.

[16] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, jul 2012.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.