Computational Intelligence ( )

Submission Assignment 5

Instructor: Anil Yaman, Bob Borsboom *Name:* Amee Tran, *Netid:* 2731525

1 Introduction

Computational Intelligence is recognised as a prevailing evaluation method for the evolution path of human (Kulkarni et al., 2022). The journey has flourished from the commencement of artificial neural network (ANN). The construction of ANN is a familiar simulation of human brain which facilitates the learning and generalisation of computer about the reality (Jain et al., 1996). However, the application of ANN or its variations has not completely derived a wealth of benefits for particular tasks. One considerable counterexample for the intuition that Convolution Neural Network(CNN) is the most efficient model for image processing has been demonstrated (Liu et al., 2018). In the paper, researchers indicated the underlying cause of the problem was lied in the deficient arrangment of the convolution layer in the feature extract for the MNIST dataset. Hence, the alternatives of CoordConv layers for the conventional convolution layer were established and the corresponding results have shown a 24% improvement over the traditional models. In addition, another study (Otchere et al., 2021) was obtained to prove the inconsistency of ANN in reaching the state of art. The experiment proposed several sophisticated ANN models with the aim to address the difficulties in the hyperparameter tunning and the objective function approximation. Nevertheless, under the scope of the limited amount of economical data, such complicated models presented an inferior performance over Support Vector Machine (SVM). Consequently, the optimisation of topology in ANN architecture has raised a significant concern to many researchers and businesses in the neural network settlement since the experiments for the efficiency of the models are computationally costly, along with arduousness.

However, various biologically-inspired characteristics, relevant to the human evolution, are validated to assist the topology analysis of ANN (Nolfi et al., 1994). More specifically, neuroevolution is one of the most potential accoutrements for deep neural network architecture (Lehman and Miikkulainen, 2013). The method is inspired from the existence of natural biological human evolution (evolutionary algorithm) in the exploration of the optimal multi-layer perceptron. Familiar to reinforcement learning, neuroevolution is driven by the reproductive fitness of a population of deep neural networks to find appropriate behaviours. Moreover, neuroevolution also generalises an enormous number of network architectures to solving the overfitting problem. As a result, it is proposed as the main method to determine the answers to the research question.

2 Problem statement

The task is implemented in the load_digits dataset. It is an open source of a 8x8 grayscale image dataset, which concerns the digital digits(Paper and Paper, 2020). The data encompasses 10 different categories with 1797 images points and, for each set, there is a balance of 180 images for both training and testing. The given responsibility is implementing the artificial neural network to classify items in the dataset and the expected outcome is a dependable accuracy in the testing set. Besides, the maximization of the accuracy is altered by minimizing the negative log-likelihood loss (cross-entropy function) between the actual values and the predicted values.

However, the configuration of ANN is extremely complex and has a countless number of possibilities to conclude which architecture should be the most appropriate model for the dataset. In another word, one can not conclude how deep the network should be and which layer should be established to prevent the feature loss of the images. Then, the problem is that the search space is currently a high dimensional environment but the aforementioned elements are non-differential. Thus, it is impossible to apply backpropagation to explore the dimension where the loss drops. Furthermore, the exploration over full model space is not scalable and apparently arduous since the traninig and evaluation for each neural network takes a considerable amount of time. As a result, this study aims to answer the following question: "Which topology is the most appropriate arrangement for the construction of deep neural network of the **load_digits** dataset to optimize the negative log likelihood function?"

3 Methodology

The implementation of the study is described in the bottom-up outline where the deep neural network is the core model and evolutionary algorithm is a primary framework. Additionally, due to the power limit in GPUs, the study would totally prioritize the horizontal development of the ANN based on the defined set of available layers and their corresponding hyperparameters. Thus, the paper would not present the evolution which relates to the depth of the network.

3.1 Deep Neural Network and Encoding Model Representation

The Neural Network is declared as a Convolutional Neural Network (CNN) with a fixed number of layers. The first option in the sequential model is a Convolutional layer $Conv2d$, followed by a non-linear function $f1(x)$. Nevertheless, image processing always estimates an expensive computational cost, the abstraction with the feature preservation is on urgent demand. Hence, the pooling layer ($Pooling$) is accordingly stacked with the aim of capturing the essentials inside a figure. Afterwards, the image is flattened into a single vector for the further classification task. The artificial neural network learns the extracted features in 2 hidden layers ($Linear1$ and $Linear2$) and the intermediate layer is another activation function $f2(x)$ to eliminate linear factors of the resulting discriminator. Ultimately, the output of the hidden layers is provided to a softmax layer for probability computation over classes. The softmax layer is defaultly set as the log softmax. Being restricted in the aforementioned conditions, the model is represented by a list of some crucial elements describing the structure of layers. As all networks utilize the same structure of the flattening layer and the second hidden layer ($Linear2$), these elements are not researched under the scope of paper. Consequently, the resulted vector comprises of 5 parts which are $Conv2d$, $f1$, $Pooling$, $Linear1$, and $f2$. The outcomes return the test loss, the classification error (ce), and the number of trainable parameters, which be manipulated to compute the fitness of a model based on the formula 3.1 where $\lambda = 0.01$, $N_p =$ the number of weights in a model, and $N_{max} =$ the maximum number weights of a model in one population.

$$fitness = ce + \lambda * \frac{N_p}{N \max} \tag{3.1}$$

3.2 Evolutionary Algorithm (EA)

Algorithm 1 General Evolutionary Algorithm

Initialise a population of solutions $\mathcal{P}_t := \mathcal{P}_0$ and evaluate
while *stop_condition* $= 0$ **do**
1. Select parents $\mathcal{S}_t$ for mating
2. Apply recombination and mutation in $\mathcal{S}_t$ to create offsprings $\mathcal{S}_{t+1}$
3. Evaluate $\mathcal{S}_{t+1}$
4. Select $\mathcal{S}_{t+1}$ from $\mathcal{S}_{t+1}$ and $\mathcal{P}_t$

end while

In Algorithm 1, a population of N CNN models, first, is generated randomly based on the established set of layer hyperparameters. Simultaneously, the evaluation is applied in the population and the individual gaining the highest fitness is declared as the best configuration. Within the loop of evolution, a more limited group of parents is specified for the mating process. The parent selection possesses a special meaning to the whole evolutionary algorithm since it directly influences the outcomes and the speed of convergence(Jebari and Madiafi, 2013). Normally, parent selection is implemented based on the fitness of individuals; however, the approach imposes a selective pressure over models possessing high-valued fitness **??**. To address the problem, according to Jebari and Madiafi (2013), the paper would research particularly tournament selection with the accompanied linear ranking. The linear ranking method is formulated in 3.2 and the general process is described in Algorithm 2. The suggested approach is believed to eliminate the dominance in population and, as a whole, decrease the probability of premature convergence.

$$\rho_i = \frac{1}{n} * (sp - (2sp - 2)\frac{rank_i - 1}{n - 1}) \quad sp \in [1, 2] \tag{3.2}$$

Algorithm 2 Parent Selection

Require: $population = \{net_1, net_2, ..., net_N\}$
$parent\_pool \leftarrow 0$
Sort individuals based on their fitness and get probabilities from the linear ranking
while $len(parent\_pool) \neq k$ **do** ▷ k is number of parents for mutation and crossover
1. Pick s individuals randomly with replacement ▷ s is number of individuals for a tournament
3. Pick the best one and add it to $parent\_pool$
end while

Subsequently, there are 2 main mating operators which are recombination and mutation, but there are possibly several variants of such combined operators.

Since the model representation is combinatorial (neural networks and integers), stochastic one-point crossover and random perturbation are implemented as the significant breeding. In terms of single-point crossover, it is the gene exchange between 2 parent chromosomes at a random pivot (Shafiee et al., 2016) (depicted in 1). From the centromere, all the genes to the left (or right) are traded between parents to produce two other children. Some studies investigated the practical performance of one-point crossover. One result (Spears and De Jong, 1991) presented that single-point crossover was superior to other forms if the length of representation is under the small scope. Thus, it is believed to culminate in a positive effect in EA.

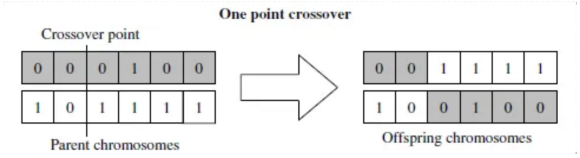

Figure 1: One-point crossover

The other operator, stochastic mutation, is primarily responsible for managing the trade-off between exploration and exploitation (Palmes et al., 2005). The underlying idea is lying under exchange one gene with another different gene in a chromosome. However, the mutation rate depends on the diversity of the population. For more details, the schema is described thoroughly in Algorithm 3.

Algorithm 3 Stochastic Mutation

Require: $Parents = \{net_1, net_2, ..., net_k\}; bounds$
$\mu = 0$ ▷ parameter controls the balance of exploration and exploitation
$i \leftarrow 0$
while $i \neq k$ **do**
  for $gene \leftarrow 0$ to 5 **do**
    $m \leftarrow random(0, 1)$
    if $m > \mu$ **then**
      $net_i[gene] \leftarrow random(\text{bounds[gene]})$
    end if
  end for
end while

Eventually, the new population is derived by the $(\mu + \theta)$- selection. The method consolidates the off-springs produced from breeding and the old parents to conclude the μ best individuals for the next generation.

4 Experiments

- Up to 1 page.
- Provide a detailed description of your experiment, e.g., what are the hyperparameters, what are their values, a neural network architecture, learning step value.
- Provide goals of the experiments, i.e., what you are going to verify.

The evolutionary algorithm will be executed in 100 generations and, for every iteration, a population of 20 individuals is produced. As mentioned in section 3, the model representation comprises of information of 5 important layers. For the Convolution layer $Conv2d$, there are 6 possibilities:

- Number of output layers: 8, kernel=(3,3), stride=1, padding=1
- Number of output layers: 8, kernel=(5,5), stride=1, padding=2
- Number of output layers: 16, kernel=(3,3), stride=1, padding=1
- Number of output layers: 16, kernel=(5,5), stride=1, padding=2
- Number of output layers: 32, kernel=(3,3), stride=1, padding=1
- Number of output layers: 32, kernel=(5,5), stride=1, padding=2

For both activation functions $f1$ and $f2$, there are 5 non-linear functions which will be investigated such as ReLU, sigmoid, tanh, softplus, or ELU function. Relating to the pooling layer $Pooling$, it is apparently a 2x2 kernel with average or sum extraction, or no applied filters. Lastly, the number of output neurons $Linear1$ for the fist hidden layer is ivestigated in range from 10 to 100. In terms of mating phase, when opting for the potential parents, the slop of the linear ranking is assigned to 1.5 and, for each tournament, 2 parent chromosomes are randomed drawn from the population and the higher-valued individual would be selected for the matching pooling. The recombination occurs in an arbitrary centromere with the fixed rate of trading but the chance of applying mutation in a single gene of a parent body is 0.5. Ultimately, the new population of 20 individuals are generated from the combination of both parent set and children set.

In case of a deep neural network, due to the high dimension of the images, the network is learned by Adam optimizer with the mini-batch gradient descent. Each batch would consist of 64 image points and the model would be trained in 20 runs to guarantee the performance. Additionally, the learning rate is default as 1e-3 and the weight decay rate is 1e-5. In the training phase, if there are more than 20 runs showing no signs of enhancement, the training will be aborted immediately. With such setting conditions, the objective of the research is seeking for a configuration that demonstrates a significant enhancement over the models used in assignment 4. In order to verify the hypothesis, the optimal construction that is received from the evolutionary algorithm would be trained in other 100 isolated times to validate the stability. The similar approach is imposed in the other 2 models being experimented in assignment 4, which are multilayer perceptron (MLP) and 2-phase convolutional neural network. Afterwards, the conclusion for the significant difference is drawn between models by statistical figures.

5 Results and discussion

- Up to 2 pages, but it could be more.
- Provide all results (plots and tables).
- Discuss all results and check whether goals are achieved.
- Compare methods (if applicable).
- Comment on deficiencies and advantages of the proposed/used methods and think of possible extensions.

Generation	Best fitness values
0	0.06174
10	0.06174
20	0.05912
30	0.05912
40	0.05912
50	0.05148
60	0.05148
70	0.05148
80	0.05148
90	0.04883
99	0.04883

Table 1: The best fitness values over 100 generations

5.1 Results

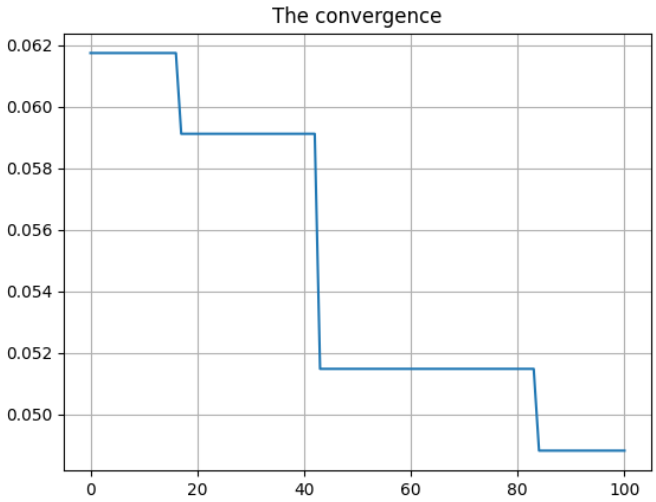

Figure 2: The convergence of evolutionary process

Figure 2 describes the speed of convergence of the evolutionary algorithm. More detailed data is presented in Table 1. The evolution is proceeded from the initial population to the 99^{th} generation. In the first 18 iterations, the best fitness rate kept unchanged. In the next generation, a slight drop was witnessed from 0.062 to 0.059 and the state was preserved in around 15 generations. Nevertheless, in the beginning of the 40^{th} evolution, the population observed a dramatic decrease of $13,5\%$ in the best individual and another change was also witnessed in the 83^{th} generation where the highest fitness is 0.048.
The outcomes of the most optimal model over evolutionary generations is depicted in Figure 3. The negative log likelihood loss of the validation set decreased gradually from 2.0 to slightly below than 1.0 and the testing set (test loss) gained a more impressive loss at 0.903. Considering the classification error (test error), the model saw a significant decline of 90% within only 6 epochs and keep fluctuated around 5% for the rest period. When compared to the accuracy of the multi-layer perceptron (MLP) and the convolutional neural network (CNN) in assignment 4, the EA model performed the greater efficiency. Precisely, in the last experiement, MLP accomplished the negative log likelihood loss at 1.102 and the classification error at 0.443. In terms of CNN, the negative loss was 0.823 and the classification error was 0.233. However, the given information is not adequate evidence, the configurations were trained in 100 isolated times to compute the mean and the standard deviation. The classification error and the standard deviation of 3 models are depicted in Table 2 and the stability of models are plotted in Figures 4, 5, and 6. In table 2, the EA model possessed the lowest classification error at 0.0723 and MLP is the worst model with the classification of error as 0.397. Similarly, EA model also gained the lowest standard deviation but CNN had the highest range of fluctuation. By inspection,

Model	Classification Error	Standard deviation	Overlap
EA model	0.0723	0.01289	NO
MLP	0.397	0.16664	NO
CNN	0.2915	0.1749	NO

Table 2: The classification error and standard deviation of 3 models

there was no overlapping between models; thus, the difference between models is significant. Consequently, EA configuration is the optimal network among the suggested models.

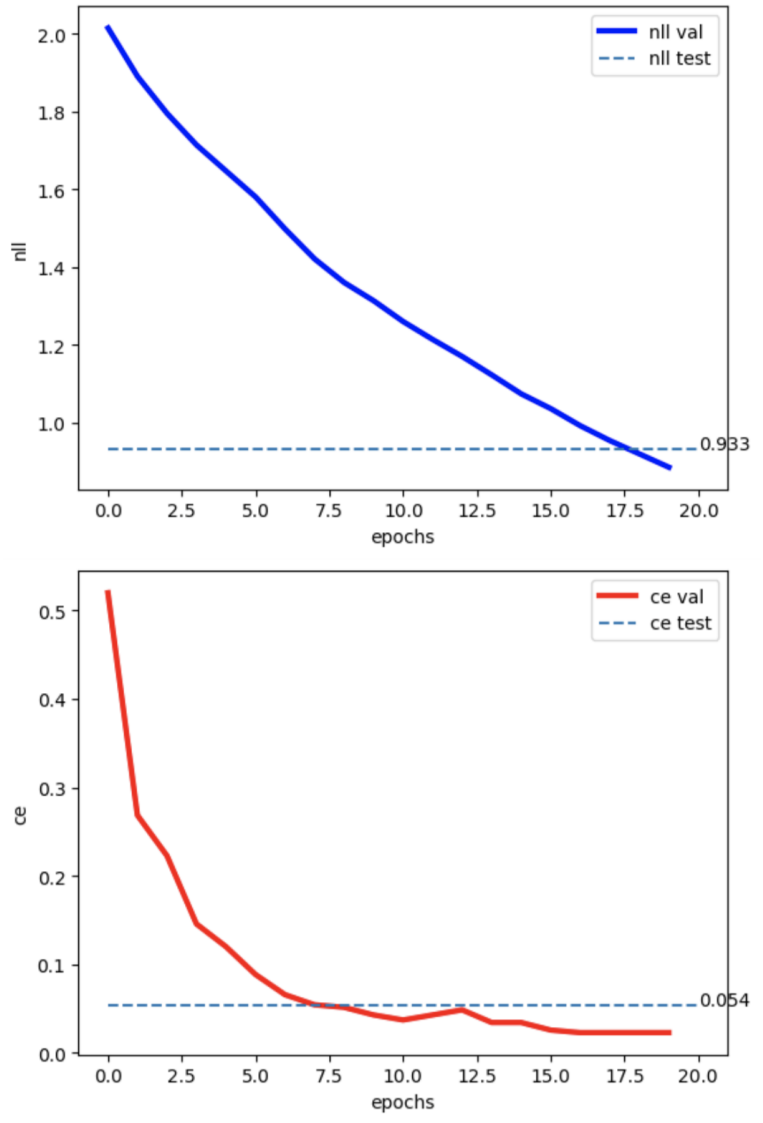

Figure 3: The test loss and the test error of the optimal

5.2 Discussion

In general, the study witnessed a gradual drop of approximately 20% in the fitness values over 100 generation. According to the graph 2, the evolutionary process did not violate the patience condition; thus, it was assumed in the continuous progress of improvement. However, the restriction of the research are set under the capacity of GPU usage and time allowance, the final result at the last run is regarded as the convergence of the evolution. In another word, it places a promising result for future project with a more powerful computational capacity. Furthermore, the test loss and test error indicate the generalisation of the model in the interaction of the reality. Nevertheless, due to the monotony of the dataset, it is insufficient to make a dependable conclusion

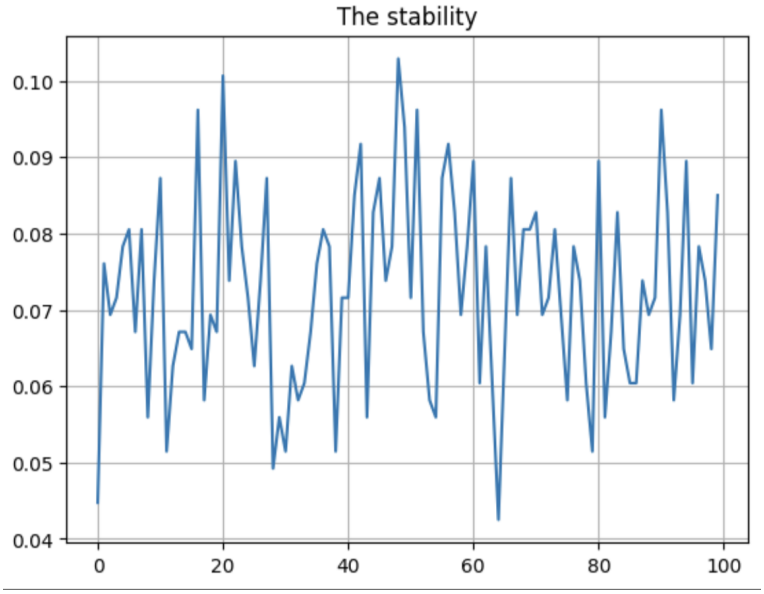

Figure 4: Stability of the evolutionary model

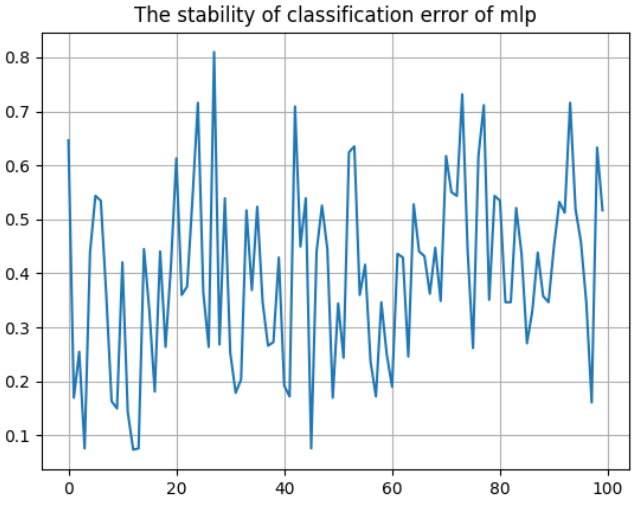

Figure 5: Stability of MLP in assignment 4

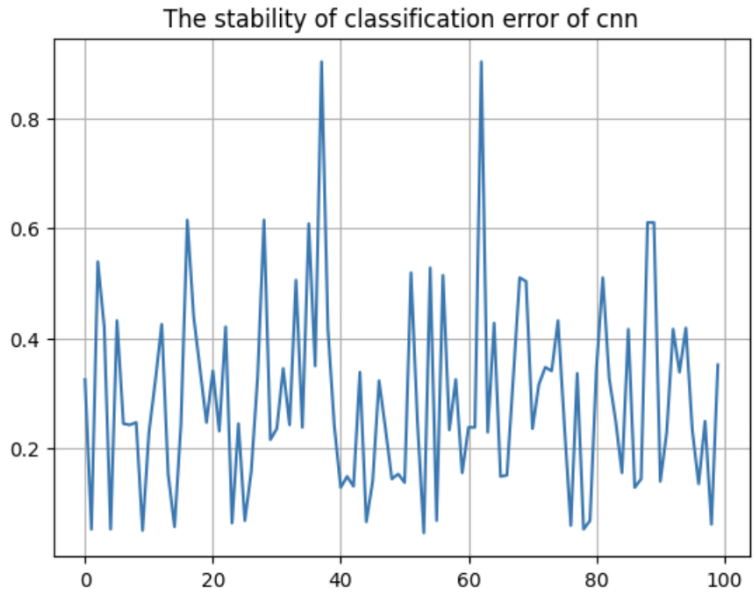

Figure 6: Stability of CNN in assignment 4

for the previous determination. As a result, a more diverse dataset is comparatively essential to promote the validity of the research result.

In the comparison between models, stability and several statistical figures are proposed as an approach to evaluate the efficiency. The MLP had 2 sequential stacks of one linear layer followed by one ReLU function and the CNN model has 2 phases of convolutional layer accompanied with non-linear activation layers and 1 deep ANN for classification task. Hence, after the comparison with CNN, EA configuration is not complicated but is the most effective one. The concept is slightly contrasting to the intuition but one reason is possibly the complicated structure of CNN model is not reasonable and its settings is inappropriate. Moreover, the figures relevant to classification error and standard deviation proved the less bias and less variance of the model. Specifically, the model is stable and less prone to random effect.

Ultimately, the research is believed to place a foundation of the application of evolutionary algorithm in the topology-analysis of deep neural network. Many perspectives of the evolutionary algorithm were investigated and some statistical methods were implemented for the valid evaluation. Additional, the comparison of models promoted the effectiveness of the evolutionary algorithm and drive more research motivation corresponding to the associated field. However, the statements derived in the study is not fully academically verified as the dataset provides partially accurate reflection; thus, the configuration of the model would be affected at some extent. Furthermore, evolution was not trained in the infinite period as inadequacy of GPU power, so the assumption of the convergence is not rational under the scope of the academic research. Consequently, the study has a valuable meaning to the intelligence of human and computer.

References

Indicate papers/books you used for the assignment. References are unlimited. I suggest to use `bibtex` and add sources to `literature.bib`. An example citation would be Eiben et al. (2003) for the running text or otherwise (Eiben et al., 2003).

References

Eiben, A. E., Smith, J. E., et al. (2003). *Introduction to evolutionary computing*, volume 53. Springer.

Jain, A. K., Mao, J., and Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. *Computer*, 29(3):31–44.

Jebari, K. and Madiafi, M. (2013). Selection methods for genetic algorithms. *International Journal of Emerging Sciences*, 3(4):333–344.

Kulkarni, A., Shivananda, A., and Sharma, N. R. (2022). The building blocks of computer vision. In *Computer Vision Projects with PyTorch: Design and Develop Production-Grade Models*, pages 1–41. Springer.

Lehman, J. and Miikkulainen, R. (2013). Neuroevolution. *Scholarpedia*, 8(6):30977.

Liu, R., Lehman, J., Molino, P., Petroski Such, F., Frank, E., Sergeev, A., and Yosinski, J. (2018). An intriguing failing of convolutional neural networks and the coordconv solution. *Advances in neural information processing systems*, 31.

Nolfi, S., Parisi, D., and Elman, J. L. (1994). Learning and evolution in neural networks. *Adaptive Behavior*, 3(1):5–28.

Otchere, D. A., Ganat, T. O. A., Gholami, R., and Ridha, S. (2021). Application of supervised machine learning paradigms in the prediction of petroleum reservoir properties: Comparative analysis of ann and svm models. *Journal of Petroleum Science and Engineering*, 200:108182.

Palmes, P. P., Hayasaka, T., and Usui, S. (2005). Mutation-based genetic neural network. *IEEE Transactions on Neural Networks*, 16(3):587–600.

Paper, D. and Paper, D. (2020). Introduction to scikit-learn. *Hands-on Scikit-Learn for Machine Learning Applications: Data Science Fundamentals with Python*, pages 1–35.

Shafiee, A., Arab, M., Lai, Z., Liu, Z., and Abbas, A. (2016). Automated process flowsheet synthesis for membrane processes using genetic algorithm: role of crossover operators. In *Computer Aided Chemical Engineering*, volume 38, pages 1201–1206. Elsevier.

Spears, W. M. and De Jong, K. A. (1991). An analysis of multi-point crossover. In *Foundations of genetic algorithms*, volume 1, pages 301–315. Elsevier.