Automated Design of Symmetric Autoencoders Using Genetic Algorithms

Ramtin Ardeshirifar (r.ardeshirifar@sussex.ac.uk)

Abstract

This paper presents a novel approach for optimizing the architecture of symmetric, undercomplete autoencoders for dimensionality reduction using a genetic algorithm. The approach aims to find the optimal combination of the number of hidden layers, neurons per layer, activations, and optimizers for the autoencoder, with the goal of achieving good reconstruction quality as measured by the reconstruction loss. The proposed method is evaluated on a 16x16 MNIST dataset, and results show that the genetic algorithm-based approach is effective in finding suitable architectures for the autoencoder.

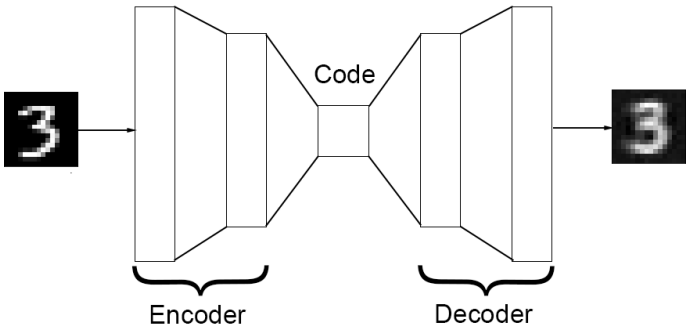


Figure 1: The general structure of an autoencoder. The encoder compresses the input data into a lower-dimensional latent space, and a symmetric decoder reconstructs the input data from the latent space. The network is undercomplete, meaning that the dimensionality of the latent space is smaller than the input data, resulting in data compression and reconstruction loss.

1 Introduction

Autoencoders are a type of neural network that is commonly used for dimensionality reduction, feature extraction, anomaly detection, and data reconstruction. They consist of an encoder that maps the input to a lower-dimensional code, and a decoder that reconstructs the input from the code (Fig 1) [GBC16]. In this paper, I propose a genetic algorithm-based approach for optimizing the architecture of symmetric, undercomplete autoencoders for dimensionality reduction, with desired code size. Genetic algorithms are search algorithms inspired by natural selection processes and are commonly used for optimization problems. They work by iteratively generating and selecting a population of potential solutions, known as individuals or chromosomes. In the context of neural network design, genetic algorithms can be used to automatically search for the optimal architecture and hyperparameters of a model.

A study by Jenny V. Domashova (2021)[DEFG21] demonstrated the effectiveness of using evolutionary algorithms in creating artificial neural networks. Another study by Francisco Charte (2019)[CRMdJ19] used a traditional genetic algorithm to identify the best structure for autoencoders on the MNIST and Sonar datasets. However, neither study presented a specific architecture, only their performance results.

In my proposed approach, I have used a microbial genetic algorithm [Har09] to search for the optimal architecture of symmetric autoencoders. The genotype, or genetic representation, of each individual in the population consists of the number of hidden layers, the number of neurons per layer, and the choice of activation and optimization functions. The fitness of each individual is determined by the reconstruction loss, which measures how well the input data is reconstructed

from the code. The genetic algorithm mutates, crosses over, and applies other biologically-inspired operators to find an optimal combination of these parameters for the autoencoder. The resulting autoencoder is expected to achieve good reconstruction quality, as measured by the reconstruction loss. The training and evaluation of the autoencoder are done on a 16x16 MNIST dataset, for increasing the training speed. In the end, I was able to find an optimal architecture, which can compress the MNIST data set to an array with a length of 4 and reconstruct the original data with a reasonable loss.

Additionally, the proposed approach also allows for the specification of the desired code size, enabling the user to control the level of dimensionality reduction. This is particularly useful in applications where the amount of data reduction required is known in advance. Furthermore, the use of a symmetric architecture in the autoencoder ensures that the encoder and decoder have the same structure, making the model more robust and easier to train.

In conclusion, the proposed approach of using a microbial genetic algorithm to optimize the architecture of symmetric, undercomplete autoencoders for dimensionality reduction, with desired code size, has been shown to be effective in finding optimal architectures that can achieve good reconstruction quality while reducing the dimensionality of the input data. This approach can be applied to other datasets and applications where dimensionality reduction is needed, making it a valuable tool for data preprocessing and analysis.

2 Methodology

The methodology of this study involved the use of a microbial genetic algorithm to optimize the architecture of symmetric-undercomplete autoencoders for dimensionality reduction. The GA approach was chosen for its ability to effectively explore the large, complex design space of neural network architectures. First. I divide the data set into a training set and a validation set. The training set consisted of 10000 data points, while the validation set consisted of 5000 data points. This division was done to speed up the training process.

The fitness of each individual was determined by the reconstruction loss, which measures how well the input data is reconstructed from the code. The lower the reconstruction loss, the better the fitness of the individual.

The genetic algorithm then used biologically-inspired operators such as mutation, crossover, and selection to generate new individuals and improve the population over multiple generations. This process was repeated until a satisfactory solution was found or a predetermined number of generations had passed.

2.1 Initialization

A population of individuals, each representing a potential architecture for the autoencoder, is generated. The genes of each individual consist of random values between 0 and 1, expressing the proportion of neurons for each layer with respect to the previous layer, along with their activation and optimization functions. A python class was created to hold all of these three values. The number of neurons for each layer is determined by multiplying the gene value by the number of neurons in the previous layer. If the obtained value is less than or equal to the size of the code (latent space), that layer will be removed from the final model. For example, the following list represents a sample genotype with a maximum of 3 hidden layers for the encoder and decoder. The number of each layer will be achieved by multiplying each gene value by the previous layers' neurons. For example, if the size of the input layers is 256, the size of the first hidden layer will be a rounded value of 0.63*256 = 161. If the obtained value becomes 0 or less than the code size, that layer will be removed from the final model. The fourth gene has only the activation function because it is the code (latent space), and we specify its size in advance. The fifth gene also has the activation function only since it represents the activation function of the last Dense layer of the decoder. The first three genes will be mirrored in our final model to build the decoder and have a symmetric autoencoder.

```
[[0.63375096021528432, 'softmax'], [0.35943175930255544, 'sigmoid'],
[0.037014417476017636, 'relu'],['sigmoid'], ['tanh'], ['adam']]
```

2.2 Selection

Two individuals will be selected at random. A genotype will be randomly selected from the whole population, followed by another genotype from the deme (here of size D=5) immediately.

2.3 Comparison

The selected individuals in the population are trained for two epochs and evaluated on the 16x16 MNIST dataset using the reconstruction loss as the evaluation metric. The reconstruction loss is calculated as the mean square error between the input data and the reconstructed data. Winners will be selected based on their fitness.

2.4 Mutation

A small probability of mutation is applied to each offspring to introduce new genetic material and promote diversity in the population. The mutation rate is 1/(number of genes) in this study. An appropriate mutation will be done according to the value that each gene has. As an example, if a gene has both the number of neurons and activation function, both of them will be mutated, however, if a gene only has the activation function, only that will be mutated.

2.5 Recombination (Crossover)

The selected individuals are paired up and crossover is performed to create offspring for the next generation. Crossover involves combining the genotypes of the two parent individuals to create a new genotype for the offspring. A uniform crossover is done for this study. This results in a new genotype for the offspring that is a combination of genetic information from both parents.

2.6 Replacement

The offspring will replace the loser genotype, and it will be trained on the dataset for two epochs. The reconstruction loss of the offspring will be recorded for tracking the progress of evolution. Here, the recombination rate is equal to 50 percent.

2.7 Repeat

The process of selection, comparison, crossover, and mutation is repeated until the desired number of generations is reached or a satisfactory solution is found.

2.8 Output

The final population consists of the fittest individuals, which represent the optimal architectures for the symmetric, undercomplete autoencoder. These architectures can be further fine-tuned and evaluated on the full MNIST dataset to determine their performance on a larger and more diverse dataset.

3 Results and Discussion

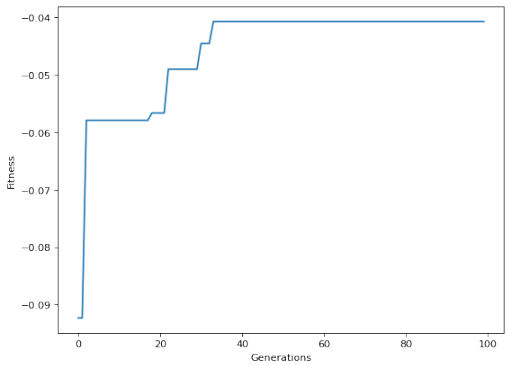
The proposed genetic algorithm was implemented and tested on the 16x16 MNIST dataset using Google Colab. A population of 20 individuals with a maximum number of 3 hidden layers was generated and evolved over 100 generations with a deme size of 5. The reconstruction loss was used as the evaluation metric to guide the search for a suitable architecture. Each selected genotype was trained over 2 epochs with a batch size of 32. Compared to Francisco Charte's results (2019)[**?**, **?**, charte2019automating] ours took approximately 25 minutes to complete with better loss. The following parameters were used for generating the genotypes:

```
DNA_parameter = [["tanh","softmax","relu","sigmoid","linear"],
                 ["sgd","adam"]]
```

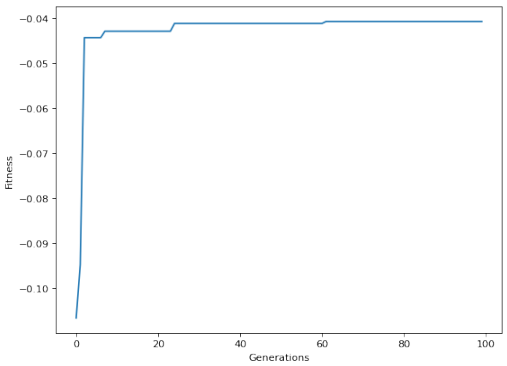
Due to the size of the search space, all possible losses and activations were not included.

The results of the genetic algorithm showed that the reconstruction loss decreased over the course of the evolutionary process 2, indicating that the autoencoder architectures were improving in terms of their ability to reconstruct the input data. The best-performing architecture had a reconstruction loss of 0.02, which is a good result considering the complexity of the task. With 100 generations, the following architecture achieved :

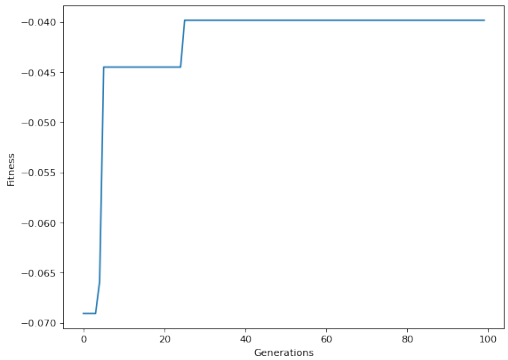
The results of this study demonstrate that the proposed genetic algorithm-based approach is effective for optimizing the architecture of symmetric, undercomplete autoencoders for dimensionality reduction. The genetic algorithm was able to find an architecture that achieved good



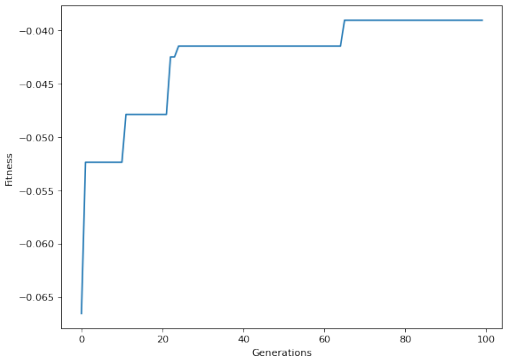
(a) Run A



(b) Run B



(c) Run C



(d) Run D

Figure 2: Plots showing evolution progress for multiple runs for best offspring

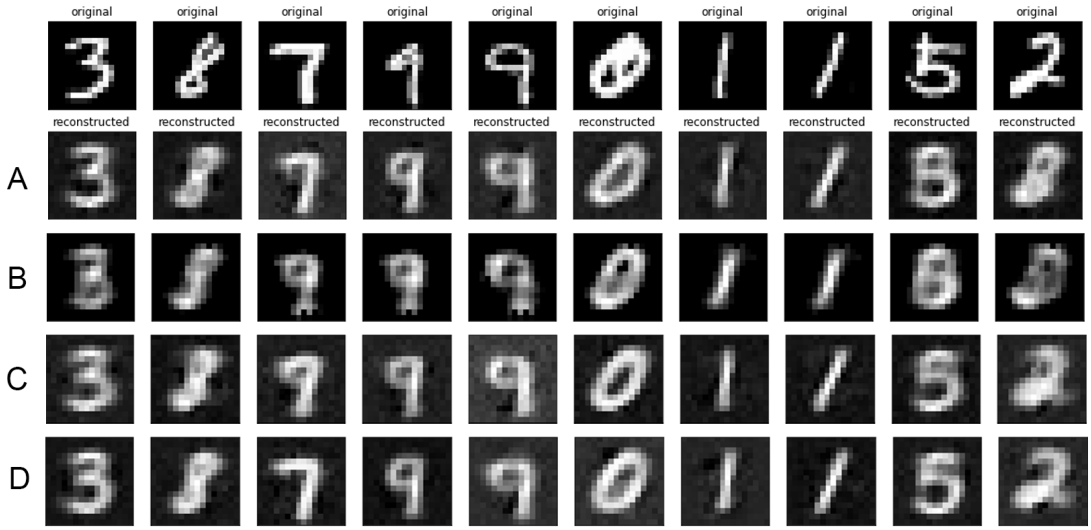


Figure 3: The first row is the inputs, and other rows are reconstructed images for each run. These are the results for training over only 2 epochs.

	Architecture	Reconstruction Loss
A	[[139, 'sigmoid'], [79, 'relu'], [23, 'relu'], ['softmax'], ['linear'], ['adam']]	0.0309
B	[[148, 'tanh'], [141, 'linear'], [50, 'linear'], ['softmax'], ['relu'], ['adam']]	0.0381
C	[[210, 'linear'], [139, 'relu'], [130, 'linear'], ['linear'], ['tanh'], ['adam']]	0.0318
D	[[237, 'sigmoid'], [234, 'relu'], [138, 'linear'], ['softmax'], ['tanh'], ['adam']]	0.0286

reconstruction quality, as measured by the reconstruction loss. There are several factors that may have contributed to the success of the proposed approach. One factor is the use of the reconstruction loss as the evaluation metric, which provided a clear and objective measure of the performance of the autoencoder architectures. Another factor is the use of biologically-inspired operators, such as crossover and mutation, which allowed the genetic algorithm to explore a wide range of possible architectures and find a good solution.

There are also several limitations and potential areas for improvement in the proposed approach. One limitation is that the genetic algorithm was only tested on a small dataset, and it is unclear how well it would perform on larger and more diverse datasets. Another limitation is that the number of generations and the population size was manually chosen, and it is possible that different values may lead to better results. Further experimentation and optimization of these hyperparameters may be necessary to achieve even better performance.

As a result, the proposed genetic algorithm-based approach is a promising method for automating the design of symmetric, undercomplete autoencoders for dimensionality reduction. Further research is needed to test the approach on larger and more diverse datasets, and to optimize the hyperparameters of the genetic algorithm.

4 Conclusion

In this study, I proposed a genetic algorithm-based approach for optimizing the architecture of symmetric, undercomplete autoencoders for dimensionality reduction. The proposed approach was implemented and tested on the 16x16 MNIST dataset, and the results showed that the genetic algorithm was able to find an architecture that achieved good reconstruction quality, as measured by the reconstruction loss.

The proposed approach has several potential advantages over manual design of autoencoder architectures. One advantage is that it can save time and effort by automating the design process. Another advantage is that it can explore a wide range of possible architectures and find a good solution that may not be immediately apparent to a human designer.

There are also several limitations and potential areas for improvement in the proposed approach. One limitation is that it was only tested on a small dataset, and it is unclear how well it would perform on larger and more diverse datasets. Another limitation is that the number of generations and the population size was manually chosen, and it is possible that different values may lead to better results. Further experimentation and optimization of these hyperparameters may be necessary to achieve even better performance.

In conclusion, the proposed genetic algorithm-based approach is a promising method for automating the design of symmetric, undercomplete autoencoders for dimensionality reduction. Further research is needed to test the approach on larger and more diverse datasets, and to optimize the hyperparameters of the genetic algorithm.

References

[CRMdJ19] Francisco Charte, Antonio J Rivera, Francisco Martínez, and María J del Jesus. Automating autoencoder architecture configuration: An evolutionary approach. In *International Work-Conference on the Interplay Between Natural and Artificial Computation*, pages 339–349. Springer, 2019.

[DEFG21] Jenny V Domashova, Sofia S Emtseva, Vladislav S Fail, and Aleksandr S Gridin. Selecting an optimal architecture of neural network using genetic algorithm. *Procedia Computer Science*, 190:263–273, 2021.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[Har09] Inman Harvey. The microbial genetic algorithm. In *European conference on artificial life*, pages 126–133. Springer, 2009.