Genetic and Evolutionary Feature Selection for Image Classification and Recunstruction

Armin Khayyer[1]

Abstract— **This project carries two main goals. First a genetic and evolutionary algorithm is used for feature selection in image classification. In fact, it is tried to find the best set of features of the images that results in a high accuracy of the classifier. for this purpose a Multi-Layer Preceptron (MLP) network is used. Secondly, once the best feature set is found, a MLP will be trained with the achieved mask, the trained model then will be used to reconstruct images for each target class using a Genetic and evolutionary algorithm (GA). That is a GA evolves a few randomly generated Gaussian noise to find an image that can be predicted as a target class with a high probability based on the trained classifier. For this purpose, The MNIST database is used. The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems.**

I. Introduction

Image classification uses artificial intelligence technology to automatically identify objects, people, places and actions in images. Image classification is used to perform tasks like labeling images with descriptive tags, searching for content in images, and guiding robots, autonomous vehicles, and driver assistance systems.

Image classification is natural for humans and animals but is an extremely difficult task for computers to perform. Over the past two decades, the field of Computer Vision has emerged, and tools and technologies have been developed which can rise to the challenge. The most effective tool found for the task for image recognition is a deep neural network. Deep learning excels in recognizing objects in images as it's implemented using 3 or more layers of artificial neural networks where each layer is responsible for extracting one or more feature of the image.

In the process of neural network image classification, the vector encoding of the image is turned into constructs that depict physical objects and features. Computer vision systems can logically analyze these constructs, first by simplifying images and extracting the most important information, then by organizing data through feature extraction and classification. Finally, computer vision systems use classification or other algorithms to make a decision about the image or part of it – which category they belong to, or how they can best be described.

In this work a MLP network is used to classify the handwritten digits MNIST dataset. The MNIST database (Modified National Institute of Standards and Technology database) is a large database of handwritten digits that is commonly used for training various image processing systems. When it comes to developing an image classifier, it is very important to know which pixels are playing a crucial role in the output of the classifier [1]. In order to find the most important features a Genetic and Evolutionary Feature selection method is used to find the best set of pixels that result in a high classifier accuracy. This is a stochastic method for function optimization based on the mechanics of natural genetics and biological evolution. In this paper a steady-state genetic algorithm has been utilized to find the best pixels or features set for the MLP model. Each individual in the population represents a MLP models. The number of genes is the total number of pixels in an image, in this case 28×28. Genes here are binary-coded, and represent the inclusion or not of particular features in the model.

The second part of this paper, tries to reconstruct a specific class of images using GA such that the pre-trained classifier predicts the true label for the reconstructed image with a high probability. The GA will start with a few randomly generated Gaussian noise, it then evolves them towards images with high probability of being a user-specified digit. The fitness of each individual is simply the softmax output of the pre-trained model.

The remaining of this paper is organized as follows: Section II explains the methodology and provides a brief literature of the researches that have been conducted in this area, Section III provides a detailed explanation of the experiments settings, Section IV provides our results, and finally Section /V concludes our work in this project.

II. Methodology

A. Multi-Layer Percepetron Neural Network

Percepetron neurons are a type of neurons that support supervised learning of binary classifiers using percepetron algorithm developed by Frank Rosenblatt [2]. In the modern sense, the perceptron is an algorithm for learning a binary classifier called a threshold function: a function that maps its input x (a real-valued vector) to an output value $f(x)$ (a single binary value):

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w}\cdot\mathbf{x}+b>0, \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where w is a vector of real-valued weights, $w \cdot x$ is the dot product $\sum_{i=1}^{m} w_i x_i$, where m is the number of inputs to the perceptron, and b is the bias.

[1]Department of Computer Science and Software Engineering Auburn University Auburn, USA

Multi-Layer Percepetron (MLP) network is a neural network consisted of layered architecture of percepetron neurons. It contains many perceptrons that are organized into layers. Unlike the early age perceptron discussed above, percepetron neurons in MLP can have different activation functions such as sigmoid functions, linear functions, ,softmax functions, rectifier linear units [3][4]. Since MLPs are fully connected, each node in one layer connects with a certain weight w_{ij} to every node in the following layer.

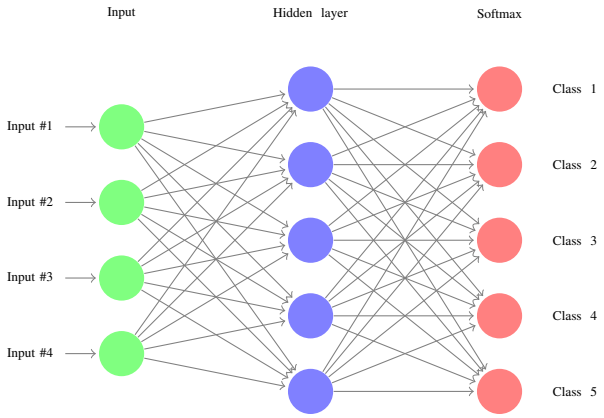


Fig. 1. Multi-Layer Preceptron Network

The degree of error in an output node j at the nth data point can be denoted as,

$$e_j(n) = d_j(n) - y_j(n) \tag{2}$$

where, d_j is the desired output value and y_j is the value produced by the percepetron. The weights at each node are targeted be adjusted such that the total error in the entire output is minimized. This error function is computed as,

$$\mathscr{E}(n) = \frac{1}{2}\sum_j e_j^2(n) \tag{3}$$

The computation of new weights at each node to minimize the error function is performed through learning algorithms. Some of the examples of training algorithms include error-back propagation (EBP) [5],scaled conjugate gradient descent (SCG) [6], levenberg-marquardt (LM) algorithm [7].

B. Genetic & Evolutionary Feature Selection

The process of finding the most relevant and significant inputs for a predictive model is called feature selection. These techniques can be used to identify and remove unneeded, irrelevant and redundant features that do not contribute or decrease the accuracy of the predictive model. One of the most advanced algorithms for feature selection is the genetic algorithm. This is a stochastic method for function optimization based on the mechanics of natural genetics and biological evolution. In this paper a steady-state genetic algorithm has been utilized to find the best feature set for the classifiers. Each individual in the population represents the predictive model. The number of genes is the total number of features in the data set, in this work the number of pixels of images. Genes here are binary-coded, and represent the inclusion or not of particular features in the model. A binary tournament selection method is used to select 2 parents from the population at each evolutionary cycle. Once the parents are chosen, a uniform crossover operator together with a mutation occurrence rate of 0.01 are used to evolve a new offspring.

In order to evaluate the fitness of each individual, the mentioned MLP methods are trained with those pixels that have a corresponding gene with "True" value, then the accuracy measures of the MLP methods over the test dataset is used as fitness of each individual. Additionally a criterion-based approach for the fitness assignment is used. Two candidate solutions x and y were compared first according to prediction accuracy on the test set, taking the larger of the two as the "winner". If the two candidate solutions had equal prediction accuracy, they were then compared according to the number of features used, taking the smaller of the two as the "winner" [8]. By doing so, any plausible tie will be broken. As the genetic algorithm is a stochastic optimization method, the genes of the individuals are initialized at random. In fact for creating the initial population, a user-specified number of individuals (otherwise known as population size) will be generated and added to the population. In this paper for generating each individual at the initialization step, for each gene, a Boolean value is selected uniformly and assigned to that gene.

C. GA Based Image Reconstruction

Image reconstruction, or image restoration, refers to recovering the original clean images from corrupted ones. The corruption arises in various forms, such as motion blur, and low resolution. Image noise refers the variations of color and brightness in an image with respect to an ideal image of the real scene. Image noise originates from the atmospheric disturbances, heat in semiconductor devices, or simply the stochastic process of incoming photons. Visually, the noise adds "dirty" grains with random intensity to the images, which in some cases severely degrades visual pleasure and image details such as edges. Image noise is ubiquitous due to lack of light, or imperfect camera sensors.

In this paper a deep learning approach is taken to tackle this problem. As mentioned earlier the MNIST dataset contains more than 60000 labeled handwritten digits. Therefore there are in total 10 different categories of images zero through nine. This labeled images are then used to train a deep neural network as a classifier. The outputs of the Deep classifier are probabilities corresponding to each digit. Simply the highest probability shows the predicted label of the classifier for the image which is passed to the model. This trained classifier is then used as a baseline model for the GA. In fact, the GA evolves a few number of corrupted images for each specific digit class to find an image which will be classified correctly by the classifier with a high probability.

Each individual in the population represents an image. The number of genes is the total number of pixels of an image. Genes here are real-coded between 0 and 255. A binary

tournament selection method is used to select 2 parents from the population at each evolutionary cycle. Once the parents are chosen, a uniform crossover operator together with a mutation occurrence rate of 0.01 are used to evolve a new offspring.

III. Experiment

As mentioned in the last section, Section II, a GEFeS algorithm is wrapped around the classifier model to evolve the best feature set (pixels) that results in the highest accuracy. Each individual of the GEFeS represents a boolean mask with length (28×28), corresponding to the number of pixels in an image, that identifies whether a pixel is used or not. Since at each function evaluation of GEFeS algorithm, a MLP classifier need to be trained over the whole dataset, the GEFeS algorithm is rather time-consuming, therefore we let the algorithm cycle for 1000 function evaluations. Also since a steady-state genetic algorithm is used and one offspring is generated at a time, in order to preserve the variety in the population, the initial population is randomly generated with 50 individuals. Moreover, a mutation rate of 0.01 is used to maintain genetic diversity from one generation of a population of genetic algorithm chromosomes to the next. Keras and Tensorflow python packages are used to code the model architecture.

Additionally, Once the classifier is trained, it will be saved, it then is used as a baseline model for the GA based image re-constructor. The GA tries to find an image that achieves a high probability of being one specific user specified digit (e.g. digit 0 through 9) by evolving a few randomly generated Gaussian noise. Each GA individual is a picture where its fitness is the softmax probability of being a target class. The GA uses a binary tournament selection with a uniform crossover operator. An initial population size of 200 together with 4000 function evaluations are used. The results are presented in the following section, Section IV

IV. Results

The following figure, Fig 2, shows the best set of pixels selected by the GEFeS method. It is worth to point out that since the digits are usually written in the center of the pictures in the MNIST dataset, the GEFeS method select the pixels that are located in the center of the images. That is, those pixels are the most important ones when it comes to the image classification.

Figure 3 shows the mean and of the frequency of pixels chosen as features during the 10 runs of the experiment. The pixels with the value of mean being 1 or close to 1 indicates that those pixels were selected the most during all 10 runs of the experiment by GEFeS. These pixels are classified as the most consistent pixels for the feature mask in order to achieve better accuracy for the classifier.

The following figure shows two type of random initialization of the GA based Re-constructor. Where first Fig.4.a shows an initial Gaussian blurred image of target class 2, the Gaussian blur is a type of image-blurring filter that uses a Gaussian function, the probability achieved from the softmax

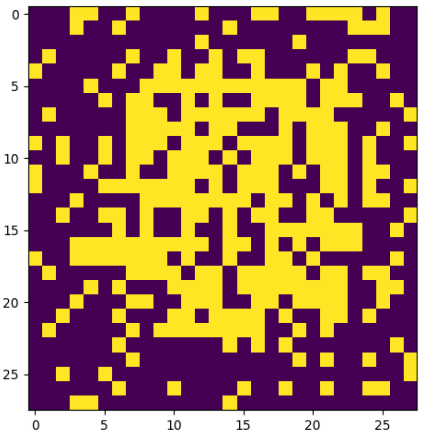


Fig. 2. A heatmap representation of the achieved mask

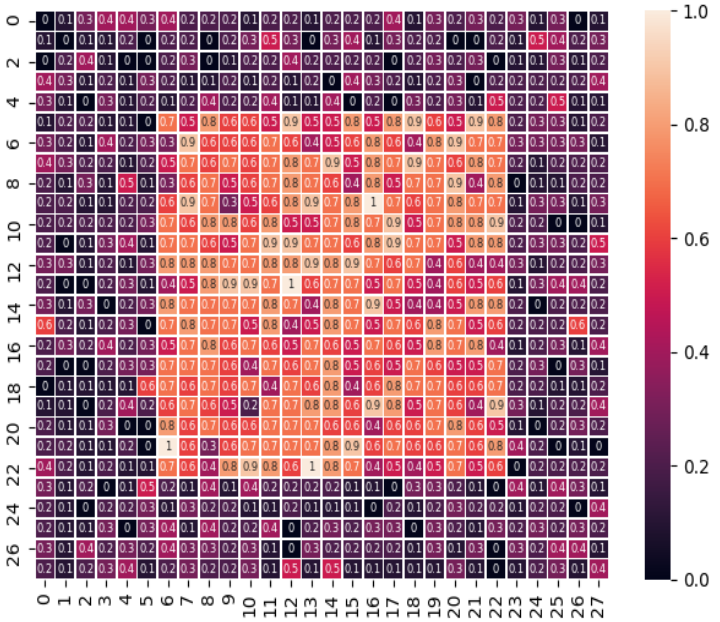
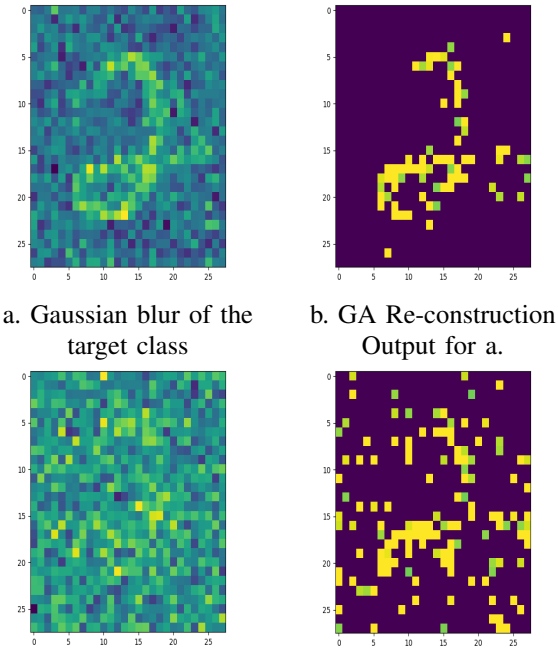


Fig. 3. Average of frequency of each pixel in the mask for 10 runs of GEFeS

layer of the classifier for this image is 0.0052, which shows that the classifier cannot distinguish if the blurred image is a 2. However, although the GA re-constructor has no notion of the baseline model, it is still able to increase this probability to 0.999997 by evolving different images. The resulted figure from the GA after 4000 function evaluation is shown in Fig.4 b.

Second, Fig.4 c shows a randomly generated image from a positive Gaussian distribution with mean of 0 and standard deviation of 255, the probability achieved from the softmax layer of the classifier is 0.001. As it was expected the results of this initialization are not as clear as that of the last method, however the GA is still capable of producing an image which is even visually distinguishable Fig. 4.d.



a. Gaussian blur of the target class

b. GA Re-construction Output for a.

c. Random Gaussian Noise $\sim (0, 255)$

d. GA Re-construction Output for c.

Fig. 4. GA Re-constructor Results

V. Conclusion

The results show that the genetic & evolutionary feature selection improves the accuracy of the MLP classifier by choosing the best pixels of the images to have as features. Additionally, the most important take away of this work is that although the GA has no notion of the problem domain, it still can find an image which not only achieve a high probability from the classifier, but also is visually clear and distinguishable.

References

[1] M. Du, N. Liu, Q. Song, and X. Hu, "Towards explanation of dnn-based prediction with guided feature inversion," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1358–1367.

[2] F. Rosenblatt, "The Perceptron—a perceiving and recognizing automaton," *Report 85-460-1*, 1957.

[3] X. Glorot, A. Bordes, and Y. Bengio, *Deep sparse rectifier neural networks*, 2011, vol. 15. [Online]. Available: http://jmlr.org/proceedings/papers/v15/glorot11a/glorot11a.pdf

[4] Wikipedia, "Multilayer perceptron - Wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Multilayer_perceptron

[5] B. M. Wilamowski, "How to not get frustrated with neural networks," *Proceedings of the IEEE International Conference on Industrial Technology*, pp. 5–11, 2011.

[6] M. Fodslette Møller, "A scaled conjugate gradient algorithm for fast supervised learning," *Neural Networks*, vol. 6, pp. 525–533, 1993.

[7] D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, jun 1963. [Online]. Available: http://epubs.siam.org/doi/10.1137/0111030

[8] K. Casey, A. Garrett, J. Gay, L. Montgomery, and G. Dozier, "An evolutionary approach for achieving scalability with general regression neural networks," *Natural Computing*, vol. 8, no. 1, pp. 133–148, Mar 2009. [Online]. Available: https://doi.org/10.1007/s11047-007-9052-x