Artificial Intelligence

Coursework I

-Prof. Tim Blackwell-

Search Algorithms and Neural Networks For Image Generation

-Report-

Mihai Ermaliuc

Contents

1. Introduction

This project performs a shallow exploration of the topic of using search algorithms in conjunction with neural networks to produce human-intelligible images. The focus is on the search methods, but a small section is dedicated to the neural networks because of their importance in the context. In the next sections the search problem will be formulated together with the objective function. Also the fitness landscape will be examined and its shortcomings identified. Then the project will show how two informed search strategies, namely First Choice Hill Climbing and Simulated Annealing could be used for the task. The implemented methods will be shown to be able to generate images consistently, although less so to induce variability between different runs. Finally, a few possible improvement directions will be enumerated.

Please note that this coursework also contains code, this paper is only the report.

2. Overview

Neural networks are widely used today for classification of images. Numerous papers and examples on the topic are freely available on the internet. The concept can be summed up in a simplistic manner as: Train a neural network so that, given an input image, it will recognize if the contents of that image belongs to a certain class (a digit, a dog, a car etc.). Basically, the network will receive at its input the image (each of its pixels) and output a number between 0 and 1 indicating the probability that the image belongs to the respective class[1].

For the purpose of generating images this project proposes the use of neural networks configured as described above in a different manner: *Given such a network and an initial image (blank, random noise etc.), gradually modify the image so that it triggers an increasingly higher output value from the neural network*. The hope is that the closer the response gets to 1 the more will the image resemble an image of that class to the human eye.

The method described above can be viewed as a problem of searching through the (input → output) space of a neural network with the goal of finding inputs (combinations of image pixels) that generate an output close to 1.

Note: the other major challenge in conjunction with the search problem is training the neural network in such a way that it facilitates the search. While this is not relevant to the topic of the coursework, it has relevance in the context of this project so a small section is dedicated to it.

3. Defining the Search Problem and Strategies

3.1. The Goal Description

This project will try to make use of the described methods and create human-intelligible images of the digit '3', *28x28* pixels in size. The base for training the neural networks is the MNIST dataset and the image size and goal have been chosen because of the public availability and large number of samples contained in this dataset, as well as for its extensive use in research for neural networks for image classification.

A little more formally, there are 2 goals:

G1 – the algorithm should be able to produce consistently human intelligible images of the digit '3'
G2 – the algorithm should be able to produce different images of the digit '3' in different runs

As it will be shown in the following sections, this project was able to reach G1 but not G2.

Note: A proper approach would have been to experiment with a variety of goals and image sizes, however this was not possible at this stage due to time constraints.

3.2. Informed Vs. Uninformed

As it will be shown below, the search space is finite but big beyond tractability (for the selected configuration the number of possible states is 256^{784}. Moreover, an objective function is available - the output of the neural network. Therefore this project discards from the start the option of using uninformed searches and focuses on the informed strategies. Due to the high depth of the search tree, informed methods that require storing paths in memory (such as Beam Search) are also discarded.

3.3. The Objective Function

This is the output of the neural network, $f(I)$ as defined in the Neural Networks section. A notable particularity of it is that the project does not consider the (0,1) output (squashed to this range by the sigmoid activation function of the final layer and used during training), but rather the value before that. This is because of the (presumed) shape of the fitness landscape and numerical rounding errors that make the sigmoid output equal to one even for small positive values of the input.

3.4. The Fitness Landscape

The fitness landscape in this case is the output of the neural network as described above. The problem of evaluating this output for deep networks in high dimensional spaces is very complex and still a topic of research. All the statements below are based on empirical observations.

For the purpose of this project, ideally there would be a value *T* such that for any output $f(I) > T$ the input is an image that satisfies the described goal (a human intelligible digit 3). During the experiments this has been observed **not** to be true. Namely, in most of the investigated cases there exists a *dominant peak* (very wide spread across the dimensions and very high in output). The corresponding input images do satisfy the goal G1, however this configuration of the landscape made the attempts at G2 unsuccessful.

The following sub-sections treat each component of the search problem definition (initial state, possible actions/transition model, goal test and path cost). Where relevant, an unconstrained formulation is presented followed by implemented and unimplemented (but possible) constraints.

3.5. The Initial State

The initial state will be the input image of size:

28 x 28 x 256 (width x height x possible colors)

The color values are integers in the *[0, 255]* range with 0 being the background color. The initial image is created by augmenting a blank image with random uniform noise. To ensure consistent evaluation between experiments a constant random seed is used.

This defines a feature space with *28 x 28 = 784* dimensions where each dimension can take 256 values. The number of possible combinations (= the dimension of the state space) is 256^{784}.

3.6. Possible Actions

Unconstrained

An action is defined by changing the value v of each pixel with an integer value in the range *[-v, 255-v]*. The resulting state will be a new image of the same size with different pixel values. This gives a number of 256^{784} possible actions from each state. It also makes the state-space graph fully connected.

Implemented Constraints

Only a limited number of pixels can change color for any given action and the change amount is limited by a small number in each direction. This ensures the algorithm will take reasonably small steps through the fitness landscape and consequently will converge faster towards a peak.

Possible (Not Implemented) Constraints

The state space and the number of possible actions from each node could be reduced by placing some restrictions such as:

- Moving to binary images (so only 2 possible values per pixel) → 2^{784}possible combinations
- Based on the training samples it can be imposed that the image must contain only a range limited number of non-background pixels.
- Further, we can place the restriction that all non-background pixels must have at least a non-background neighbor

The math for calculating the possible combinations with the last two restrictions is more involved. However, I would intuitively speculate that all these will lower the size of the search space but would not make the problem tractable for exhaustive searches.

3.7. Goal Test

The output value of the neural network is used as the goal test. The ideal case would be to declare the goal reached when the output is very close to one (i.e. > *0.99*). However, due to issues in training the neural network (see the Neural Network section) this is not feasible – there are too many false positives. Therefore, the adopted solution is to actually consider the output value before applying the sigmoid activation function in the final layer. This value will have a normal distribution centered close to 0 with a standard deviation usually lower than 100 (depending on the configuration of the neural network). The goal in this case will be to get to a value as high as possible, but without specifying any bounds.

Experimentally it was observed that the increase of the output value decreases exponentially as the algorithm progresses. So the algorithm will be stopped either after a fixed number of iterations or after this increase gets below a certain value.

4. About the Neural Networks

In order to keep this coursework focused, the report will not go too deep into the topic. However, they play two essential roles in the context: providing the objective function and modelling the fitness landscape. Neural networks used for image classification can be described as structures modelling a function:

$$f(I) = O, \quad where$$

$$I = \begin{bmatrix} I_{11} & \cdots & I_{1n} \\ \vdots & \ddots & \vdots \\ I_{m1} & \cdots & I_{mn} \end{bmatrix} is\ a\ m\ x\ n\ image$$

I_{ij} $is\ a\ pixel\ value\ (color)$

$O\ \in (0,1)$ $is\ the\ probability\ that\ the\ input\ image\ belongs\ to\ the\ class$

The MNIST dataset used for this project contains about 60,000 + 10,000 test and training samples of handwritten digits of size *28 x 28* pixels, normalized and centered. To fit the purpose, it was modified in the following way: 4430 images corresponding to digit ‘3’ were extracted and used as positive samples; 998 + 2 randomly generated (noise) images and all black/white images were created as negative samples; 3330 images of digits other than ‘3’ were randomly extracted from the dataset and also used as negative samples. Using this dataset the project created a ‘digit 3 against the world’ classification, though a larger dataset and a wider range of negative images would have probably made a positive impact.

Various architectures of networks were examined, in the end most of the investigation being performed on two types: a shallow 2-layer network and a deeper 5-layer network, with varying number neurons on the inner layers. In all cases the sigmoid activation function has been used on all layers during training. During searching however, the pre-sigmoid output of the network was considered (so a linear combination of the neurons of the previous layer). The reason for this was the rounding error that made *sigmoid(x) = 1* for values of *x* far lower than the maximum output of the network, thus failing to provide a measure of the improvement above a certain range.

In the end it could be concluded that the training procedure generally failed to create suitable landscapes for the goal G2 in the case of shallow networks, as they tend to form a single dominant peak. On deeper networks both G1 and G2 were not reached in general, as they appear to have a too rugged landscape which causes the search to get stuck. A few possible causes are the dataset being undersized (about 9000 samples for a 784-dimensional space is still a very low number) and the negative samples not being variate enough to smoothen the low-level landscape.

5. Implementation and Results

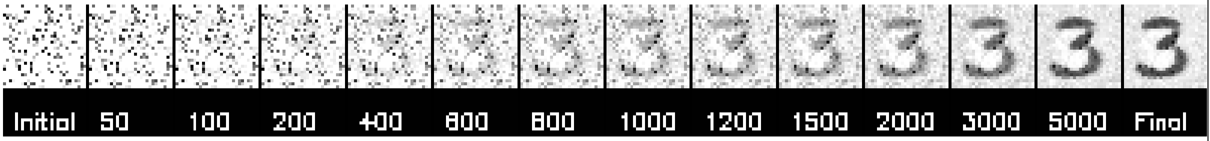
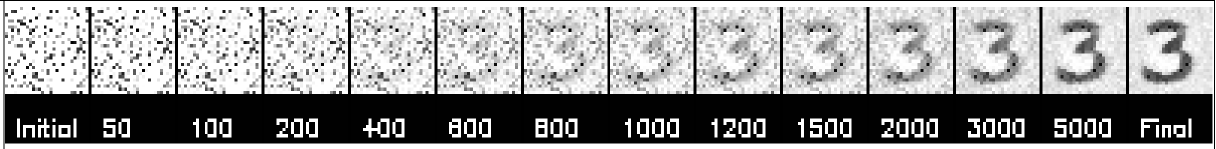
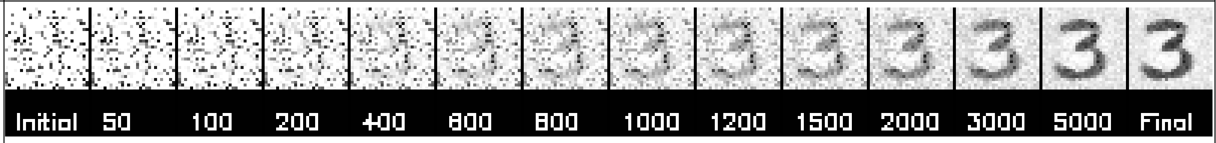
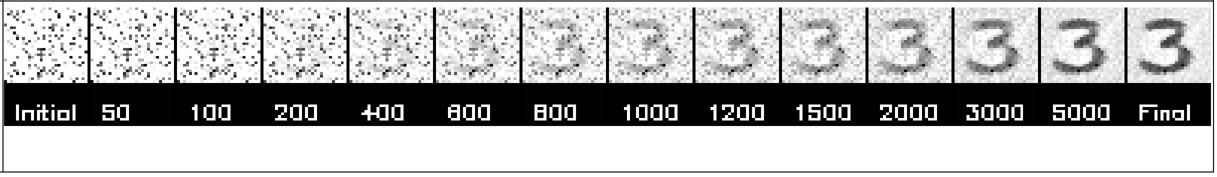
5.1. First Choice Hill Climbing

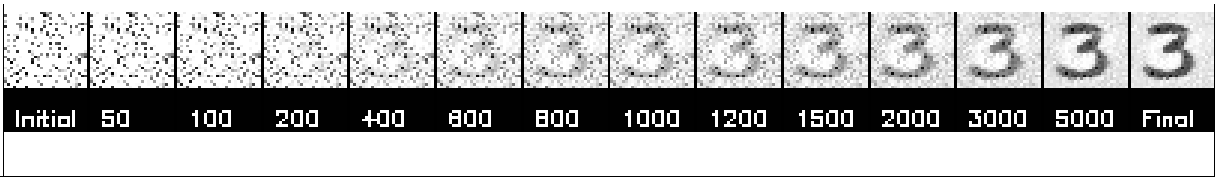
In this approach each action must be a step to a higher value over the fitness landscape, i.e.$f(I_{n+1}) > f(I_n)$. As stated above, an exhaustive expansion of all direct children of any node (to implement a simple Hill Climbing / Best-First Search) would lead to intractability. Conversely, limiting the change to a minimum (one pixel to be changed with only one unit of color) gets the algorithm stuck in meaningless local peaks. To overcome that a heuristic has been selected empirically, as follows:

- Choose 5 random pixels that will change their colors and a random color gradient for each pixel, but no more than 10% of the current color value
- If the resulting state (image) triggers a higher score from the neural network, select it
- Otherwise, repeat from the beginning

Results

Using 100000 iterations, the output shapes from various runs is shown below. The numbers below each image show the n-th step for which there was an increase in the score.

1	 Initial 50 100 200 400 600 800 1000 1200 1500 2000 3000 5000 Final
2	 Initial 50 100 200 400 600 800 1000 1200 1500 2000 3000 5000 Final
3	 Initial 50 100 200 400 600 800 1000 1200 1500 2000 3000 5000 Final
4	 Initial 50 100 200 400 600 800 1000 1200 1500 2000 3000 5000 Final

5	

Conclusions

Two important conclusions can be drawn from examining them (and the others not shown here):

a) This approach managed to produce consistently an intelligible image of the digit '3', therefore the goal G1 could be considered reached (There was actually no session for which the algorithm didn't reach the final image).

b) While all the final images are quasi-identical, by comparing those at steps 800 - 1200 between different runs it can be observed that there are slight variations in the shape of the digit. This could be interpreted as the algorithm 'climbing the hill' from different sides and does show some promise for future developments and reaching the goal G2.

5.2. Simulated Annealing

This method is an extension of the previous, with the following change: a generated step may be accepted even if it is not an improvement over the previous one, with a probability that decreases as the algorithm progresses. The general formula for determining the probability is

$$p = e^{-\frac{\Delta f}{T}}$$

where

$$\Delta f = f(I_{n+1}) - f(I_n)$$

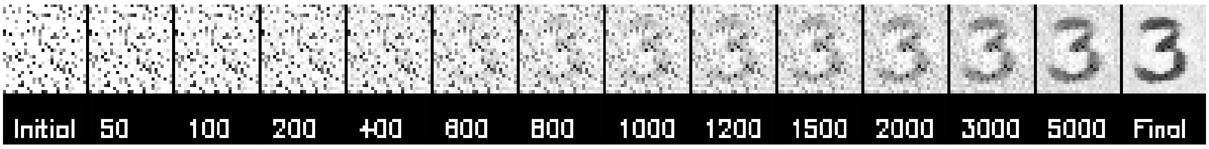
is the difference between the scores obtained by the 2 images and T is an arbitrary parameter that changes with time in such a way that p decreases with time

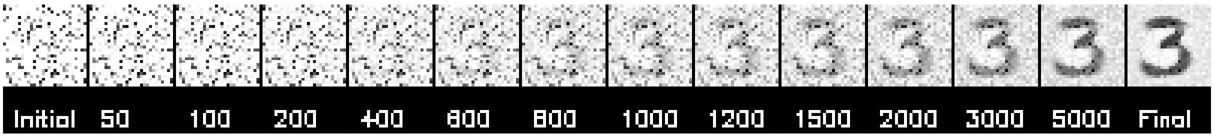
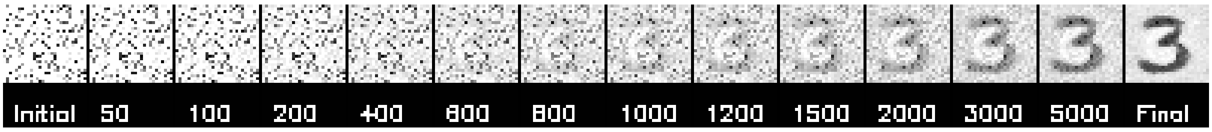
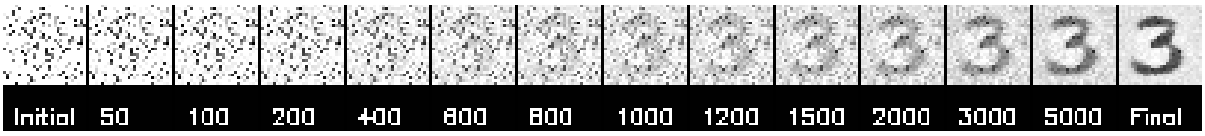
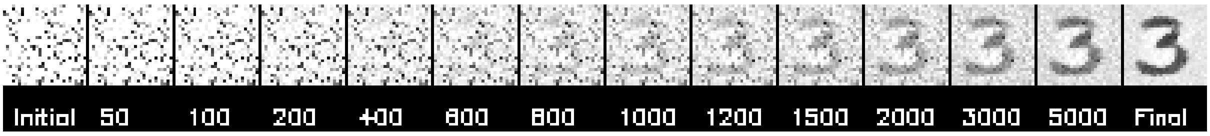
The reasoning behind using this method is to allow for better exploration of the search space during the early stages of the algorithm. The selected formula for T is:

$$T = 0.99^{i}$$

where i is the iteration number (between 0 and 100,000)

Results

1	

2	 Initial 50 100 200 400 600 800 1000 1200 1500 2000 3000 5000 Final
3	 Initial 50 100 200 400 600 800 1000 1200 1500 2000 3000 5000 Final
4	 Initial 50 100 200 400 600 800 1000 1200 1500 2000 3000 5000 Final
5	 Initial 50 100 200 400 600 800 1000 1200 1500 2000 3000 5000 Final

Conclusions

There are no significant improvements over the previous method with regard to the speed of convergence or the diversity of the images. On the contrary, the algorithm starts converging later (which is normal). It could be speculated that because of the particularities of the fitness landscape, the algorithm just “wanders around” at the bottom of the big hill until the cooling function makes it impossible to accept non-improving solutions, then it runs identically to First Choice Hill Climbing. Speculating even more, due to the random initialization of the first state this method would bring little improvement over the first one even if different peaks were easy to reach.

6. Project Conclusions

As stated in the first section, this was only a shallow exploration of the topic. The chosen test case was extremely simple, with low resolution grayscale image of a digit. The success was only moderate, with the algorithms managing to achieve human-intelligible images consistently but failing to achieve (almost) any variability.

Finally, this approach might have other uses in the context of neural networks – assessing the “quality” of the function learned by the network by trying to generate input samples that trigger incorrect responses.

Identified issues

For shallow neural networks (2-3 layers), a single dominant peak is formed in the fitness landscape corresponding to a single digit shape (at the top). This secures the generation of an intelligible image and “climbing it from different sides” generates slightly different intermediary images. However, smaller peaks representing other good solutions are so highly dominated by this one (assuming they even exist) that the methods were never able to identify one.

For deep neural networks (> 3 layers) no such dominant peak has been identified. However, the proposed algorithms do not work so well as they often get stuck on low meaningless peaks. So the search problem is more complex in such spaces, needing smarter search methods than the ones implemented here.

Possible improvement directions

For searching the space of shallow networks, methods that generate less noisy images might be more successful in achieving G2 (variability). The methods described in the *Defining The Search Problem - > Possible Actions → Unimplemented Constraints* section might actually help with this task.

For searching the space of deep networks, different heuristics for the search strategies should be identified. Intuitively, these should facilitate exploration of spaces with distant relevant peaks having multitudes of small irrelevant ones in between. Possibly better choices would be:

- Particle / Swarm Optimization
- Genetic Search
- Even First Choice Hill Climbing with frequent restarts (needs a more powerful machine than what was available for this project)

A different but equally important approach would be to look into the training of the neural networks and aim to generate better-fit landscapes.

7. References

Tim Blackwell (2017). *Artificial Intelligence Coursework*. Goldsmiths University of London

Nikolay Nikolaev (2017). *Neural Networks Coursework*. Goldsmiths University of London

Stuart Russel and Peter Norvig (2010). *Artificial Intelligence – A Modern Approach*. Prentice Hall

Yann LeCun et Al. *The MNIST Database of Handwritten Digits*. http://yann.lecun.com/exdb/mnist/

Kurt Hornik (1990). *Approximation Capabilities of Multilayer Feedforward Networks*. Technische Universitat Wien, Vienna, Austria