Master Thesis: Trapping Time in Neural Networks - a Glassy System perspective

Author: Maciej Maruszczak

Supervisor: Dr Chiara Cammarota

September 2019

1 Abstract

The connection between glassy systems (GSs) and deep neural network (DNNs) has been investigated, but some GS tools such as trapping time analysis has not been tested in DNNs setting. This thesis investigates if the trapping times along their usual training path can be obtained from DNNs to facilitate their analysis with GS methods. By following the training process which utilises stochastic gradient descent (SGD), critical points with approx. zero loss gradient, or traps, are identified by performing another SGD search aimed at minimising the loss gradient's norm G. Although in the initial learning phase all results converge to the same trap, when the loss falls rapidly distinct traps become visible and their trapping time can be analysed. The trapping time seems to increase along the training process and its distribution appears to be following power law indicating glassy behaviour. Further studies are needed to fully utilise this technique in DNN setting.

Contents

2 Main Text

2.1 Introduction

2.1.1 Background description

Glassy systems (GSs) have been investigated by researchers for more than 50 years, with a number of theoretical models describing their performance [1]. In an example of GS model, a set of randomly interacting magnetic particles attempt to minimise their combined energy by rearranging their spins [2]. This search for a minimum energy, or in other words the relaxation process, can be conceptualised as a local search for a minimum energy point on the energy landscape, with the temperature of the system adding stochastic noise to the dynamics. What makes the relaxation behaviour of the GS interesting is that they exhibit aging, or in other words the energy of these systems keep on decreasing without ever seeming to reach the minimum energy. This finding made researchers think of a number of possible models explaining this behaviour which were involving to a different extent the idea of the energy landscape being non-convex or "rough" which prevents the relaxation dynamics from quickly reaching the minimum energy [3]. In this "rough" energy landscape, the relaxation can get "stuck" in meta-stable "traps"- which are local minima which temporarily prevent the dynamics from reaching the global minimum [4]. This "trapping time" (i.e. the time at which the dynamics is temporarily trapped) can be calculated and analysed, and in the process of analysing GSs scientists have developed a range of tools, including the analysis of this trapping time distribution [5].

In parallel to the development of the GS models, Deep Neural Networks (DNNs) have become a widely acclaimed and researched tool employed in classification problems [6]. DNNs are being inspired by the neural structure of the brain and try to utilise it to automatically solve decision problems. A DNN therefore consists of a number of "neurons" which are organised in a number of "layers" [6]. These layers are arranged in series, with neurons in each layer being connected to the neurons in the neighbouring layers, with the strength of these connections being defined as weights of the DNN. The resultant structure of many consecutive neuron layers creates the "deep" structure. In order to obtain an output prediction from the DNN, the network is feed inputs on one side of the model, which are then manipulated in accordance to the weights and activation functions of the neurons in each layer before being sent to the output layer where the prediction is being made [6]. An example of a DNN is presented on Figure 1.

Before DNN can perform the classification task accurately, it needs to be trained. This involves providing input training data to a network and obtaining, at first very inaccurate, results, based on the initial random neuron connection weights. This accuracy is measured in terms of a "loss", which is a measure indicating how far the predictions of the DNN are from the correct predictions (the precise definition of the loss will be introduced later). Once the loss is established, its gradient is calculated with respect to all weights. With the known loss gradient, the weights can be updated in such a way as to minimise the loss. This process is repeated, with the loss decreasing slightly with each iteration, until it reaches the minimum loss [6].

The above algorithm, where the whole training set is used for each training step is called gradient descent (GD) as the dynamics "descents" along the loss gradient to the minimum loss [6]. Alternatively, not all but only parts of the training data, called batches, can be used to train the DNN, which makes the training computationally more efficient without much sacrifice in the final accuracy. However, due to training the network in each iteration only on a sub-sample of the training data, it may be the case the it will not follow directly along the loss gradient of the total training data, which adds a stochastic element to this method, which gives birth the name of Stochastic Gradient Descent (SGD).

Regardless of GD or SGD being used, the DNNs are minimising loss on a non-convex loss landscape: a problem which is very similar to the energy minimisation of GSs [6].

As both DNNs and GSs attempt to solve similar problem, the minimisation of a non-convex function, researchers quickly established many similarities between the two problems [8, 9]. Given the strong connection between DNNs and GSs, it seems natural to try to employ the tools which were initially developed for the analysis of GSs, such as aforementioned analysis of the trapping time [4], in the context of DNNs. The insights obtained in this manner could provide us with a different dimension on which the two systems can be compared, which could either mean further confirming the parallels between DNNs and GSs, or alternatively provide explanations for some fundamental differences between the two, such as not getting stuck in sub-optimal local loss minima in DNNs versus being trapped in local energy minima in GSs [8].

The subject of my thesis is therefore the following- to evaluate an example DNN by, in the first place, attempting to search for the meta-stable zero gradient states along the usual training process of the DNN. Once it is shown that these meta-stable states are identifiable, I will experimentally analyse the distributions of their trapping time along different phases of the training process. I will therefore demonstrate that the analysis of the trapping time of

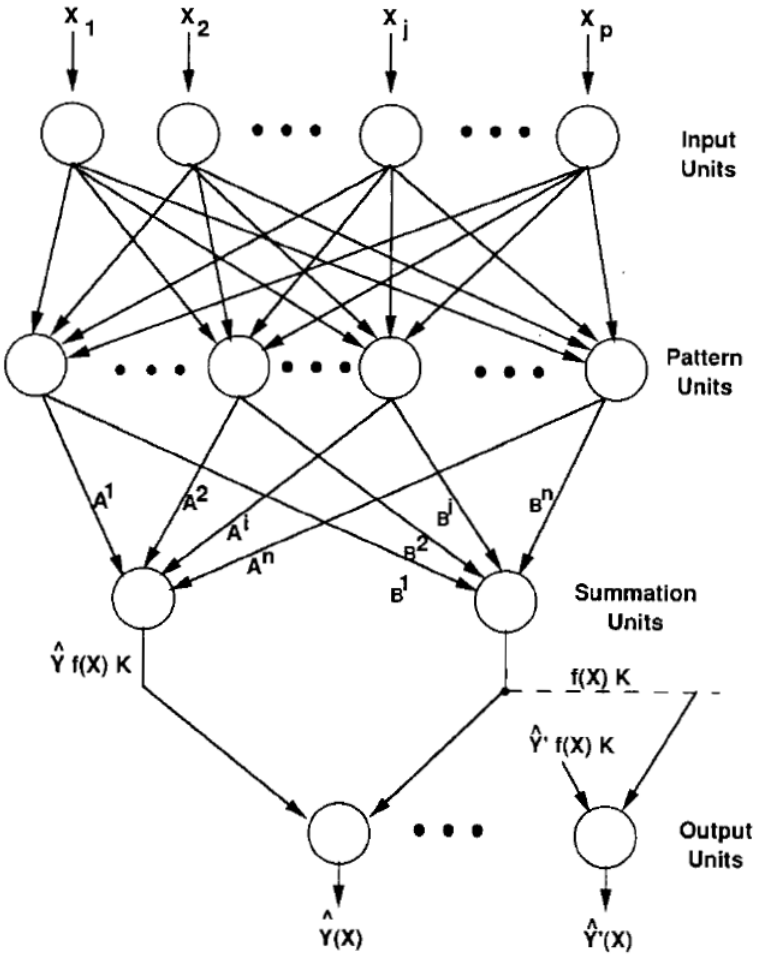


Figure 1: An example of a Deep Neural Networks, with layers of neurons clearly visible [7].

the meta-stable states which was at first developed for GSs could well be employed in the analysis of DNNs.

2.1.2 Thesis structure

The next section of my thesis will evaluate other work which was researching the trapping time in GSs, as well as meta-stable states, or in other words traps in DNNs. I will then explain the steps used in the analysis, briefly outlining the technical issues which had to be solved before the main problem of this thesis could be addressed. Once the methods are clearly laid out, I will proceed to presenting the experimental findings, which will be followed by the discussion and conclusions.

2.2 Review

2.2.1 Trapping on GSs

As mentioned previously, trap models of GSs were extensively analysed by researchers, such as Bouchaud and Dean [4]. A more realistic version of GSs, such as step models where the escape from meta states is easier (see Figure 2), were analysed by Barrat and Mézard [5].

More recently, researchers started evaluate models following more sophisticated dynamics, such as random energy models with Metropolis algorithms [10, 11]. These works outlined important features which are also clearly visible in the DNNs, such as dynamical regimes with different behaviours (see Figure 3)

Figure 2: Graph a) represents a trap model and graph b) a step model [5]. It is clear that the escape from the step model requires lower energy as the escape can be performed in steps.

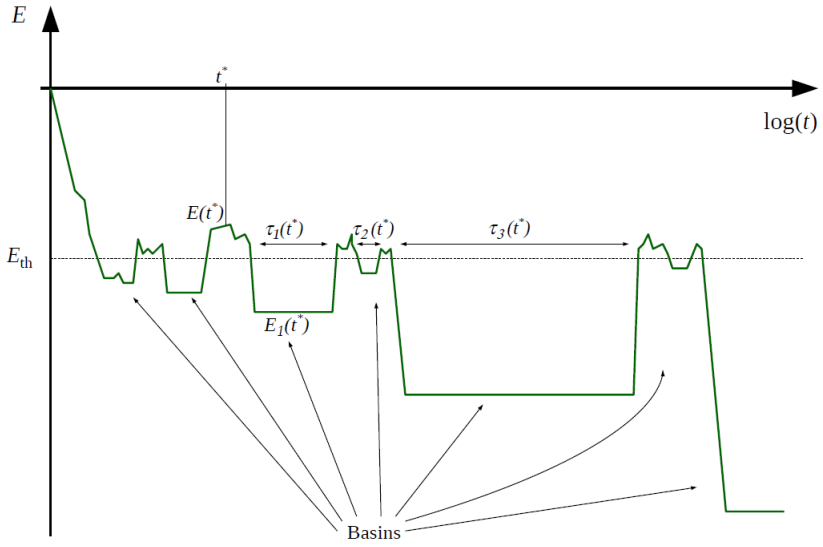


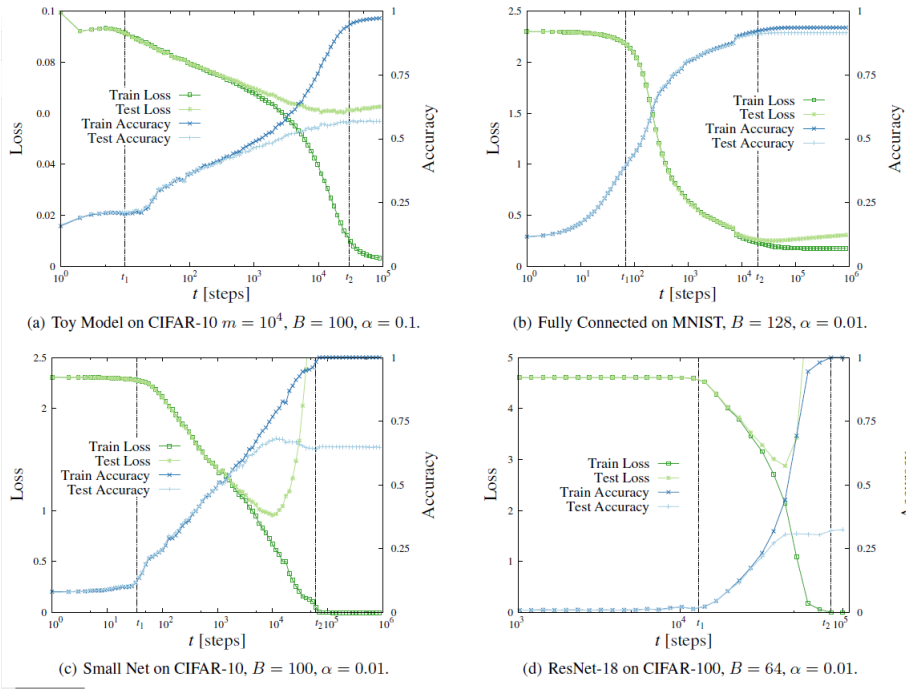
Figure 3: The evolution of energy over time, with basins indicating that the systems is in a trap and τ representing the time spent in a given trap [10]. We will later see that similar traps can be found on the training path of a DNN.

2.2.2 DNNs analysed from SGs perspective

Having established the SG trapping literature, I can proceed to the main inspiration behind my thesis- Baity-Jesi et al. 2018 paper, which analysed DNNs using the tools developed for GSs [12]. In that paper, 4 different DNNs are being trained using Stochastic Gradient Descent (SGD) and have been evaluated from a GS perspective. Baity-Jesi et al. divided the training process into three phases, where they associated the last phase, which exhibit slow convergence towards no error, with diffusion, and not the usual glassy behaviour. Nevertheless, the first and the second phase, as outlined on Figure 4 provide us with an interesting question- what would be the distribution of trapping times within them? With the first phase showing no interesting behaviour and having almost constant loss, and the second phase appearing to exhibit ageing behaviour and having a rapid loss reduction, approximately linear in log(t), the difference in the trapping behaviour between the two phases could help us confirm or identify if the fundamental differences between them do exist and if the second phase indeed exhibits aging.

2.2.3 Traps and critical points (CPs) in DNNs

The study of CPs of DNNs loss landscape, i.e. the points where the gradient of the loss function is zero, is very young branch of non-convex optimisation. Frye et al. 2019 showed that for some simple models where the underlying distribution of the CPs is known from using analytical means, the numerical methods can indeed find the correct CPs [13]. Although important as a conformation of the feasibility of the numerical search for critical points, Frye et al. was evaluating all CPs of the loss landscape, not those the closest to the path of a particular SGD training



(a) Toy Model on CIFAR-10 $m = 10^4$, $B = 100$, $\alpha = 0.1$.

(b) Fully Connected on MNIST, $B = 128$, $\alpha = 0.01$.

(c) Small Net on CIFAR-10, $B = 100$, $\alpha = 0.01$.

(d) ResNet-18 on CIFAR-100, $B = 64$, $\alpha = 0.01$.

Figure 4: The evolution of loss and accuracy in different models with three distinct learning phases indicated by the dotted lines [12]. In this thesis, I will focus on the first two phases.

process. Additionally, due to the simplicity of the models Frye had to use to have the analytical solution, they are small enough to evaluate the Hessian of the loss landscape at the CP, which would be computationally difficult to perform for the models of two orders of magnitude larger which were used by Baity-Jesi [12], which will be studied in this thesis.

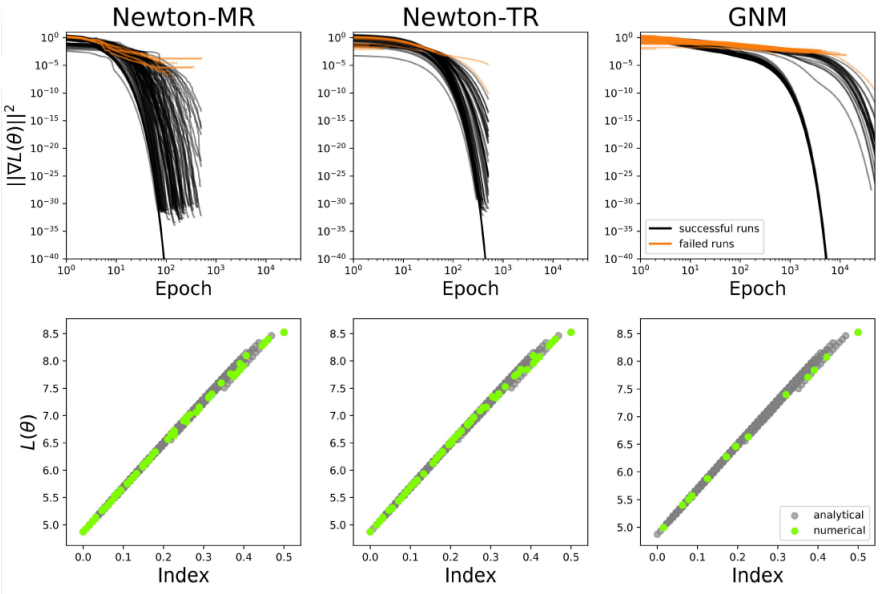


Figure 5: The three numerical methods of finding CPs evaluated by Frye et al. [13]. The rightmost method, Gradient Norm Minimisation (GNM), will be used in this thesis in the form of the Stochastic Gradient Descent method (SGD).

2.3 Methodology

2.3.1 General description and software

As outlined above, the backbone of this thesis will be the standard training process of a deep neural network, utilising stochastic gradient descent. This usual SGD algorithm, which aims at minimising the loss function will be referred to as "outer SGD". In order to ensure that the underlying outer SGD is performing correctly, I will try to replicate part of the results produced by Baity-Jesi 2018 [12].

Once the outer SGD is working well, I will search for the traps along the learning path of the outer SGD. In other words, I will search for the critical points (that is points on the loss manifold with the loss gradient of zero) which are attracting the dynamics across the learning process of the SGD. This will be performed by using another SGD algorithm, which this time will be minimising not the loss, but the norm of the gradient vector of the loss w.r.t all weights, or G function, which will be clearly defined in the coming sections. This G-minimising SGD, will be referred to as "inner SGD", as it is being performed from each individual point of the outer SGD pathway.

With both outer and inner SGDs, we can evaluate if the critical points identified along the outer SGD by inner SGDs are the same critical points, and if yes, how many points along outer SGD converge to the same CP, or trap, which would be the equivalent of the trapping time defined for GSs, but in the DNN setting.

As Frye found in their work, there are other methods which can be used in identifying the critical points, such as augmented Newton-Ralphson methods [13], which bring computational benefits when compared to SGD [13]. While acknowledging that SGD is not the most efficient method of finding CPs, as in this work I am not focused on finding any CPs but those along the path of a SGD algorithm, I decided to use the same algorithm for finding the CPs as the one used for the underlying minimisation of the loss.

Alternatively, it could have been argued that in order to find the true minimum G, employing Gradient Descent method would be more appropriate as it removes the unnecessary stochasticity stemming from using only a batch of data to update the weights, instead of the whole training data set. While in principle true, the computational burden associated with using GD would increase the calculation time by such an extent that it would render the analysis practically impossible, while SGD is performing well enough, as it will be presented later.

The aforementioned calculations are initialised in the standard procedures of the PyTorch library (version 1.1.0), following the method employed by Baity-Jesi 2018 [12].

2.3.2 Outer SGD - replication of Baity-Jesi 2019 results

After preparing the outer SGD code, the results presented on Figure 4, sub-graph b), were recreated using my code, and were used as the basis of further analysis. The data analysed was MNIST data-set, consisting of 60,000 hand written digits [14]. MNIST data-set was analysed by a simple neural network with three fully connected layers, of sizes 100, 100 and 10, respectively [12]. The non-linear functions are ReLUs, and the loss function is the negative log-likelihood of soft-max outputs [12]:

$$p_k = \frac{e^{f_k}}{\sum_j e^{f_j}} \text{ and } L_i = -\log\left(p_{y_i}\right) \tag{1}$$

Where p_k is the soft-max output for category k, f_j is the feed-forward output for category j and L_i is the loss of output i, which is calculated at the correct category and summed over all training or testing data to give the total loss. The total number of weights to be evaluated is 89,610. The model utilised batches of 128 pictures, and a constant learning rate α of 0.01.

Following averaging of the results from 8 separate runs, as in Baity-Jesi paper, the results presented on Figure 6 were obtained.

The slight truncation on the right hand side was done due to the computational requirements- as the horizontal axis is logarithmic the last part constitutes of 10 times larger computational expense. Although it was unpractical to run 8 simulations for that long to obtain their average, I have run one and its result, presented on Figure 7, indicate that my results indeed mirror those obtained by Baity-Jesi [12]. One can therefore see that the results are practically identical, giving confidence in the underlying outer SGD used in the proceeding analysis.

The analysis will be therefore based on the outer SGD presented in this section, with one minor exception. The learning rate chosen by Baity-Jesi of $\alpha = 0.01$, requires circa 10 epochs to be evaluated in order to obtain a good accuracy (circa 90 %) from the predictions of the DNN. By increasing the learning rate of the outer SGD to 0.1, the phases described in Baity-Jesi 2019 are clearly observable by evaluating just a single epoch, as presented on Figure 8.

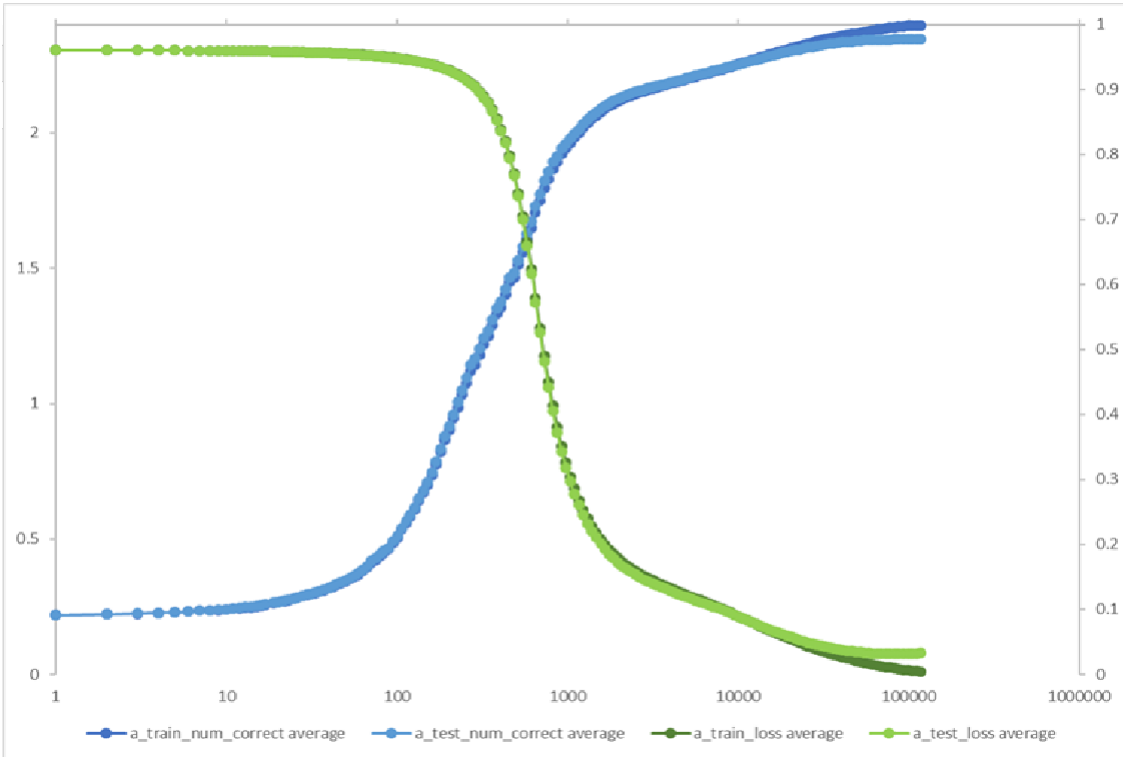


Figure 6: The results obtained from my code, which attempts to reproduce the results presented in Baity-Jesi 2018 paper [12]. The dark and bright green lines represent the loss obtained on the whole training and test data, respectively, whilst the dark and bright blue indicate the accuracy of the predictions on the training and testing data. To evaluate if my results are comparable to Baity-Jesi 2018, please compare these results to Figure 4, sub-graph b), which utilises the same colour coding.

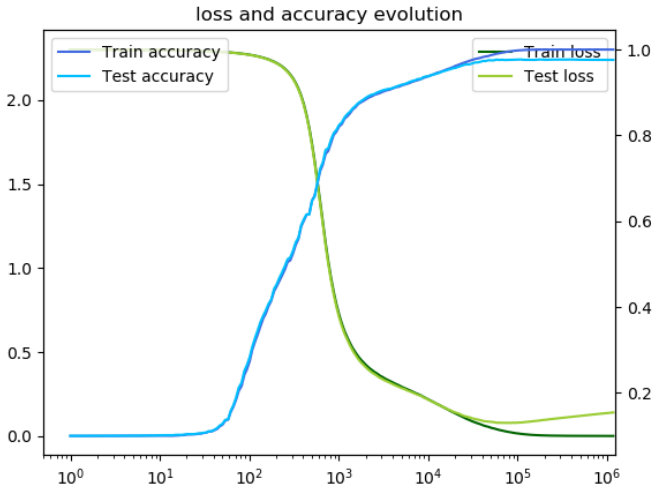


Figure 7: A single, long term run of the DNN training, indicating the agreement with Baity-Jesi results[12].

Hence, for the sake of computational efficiency, the outer SGD will be following methodology of Baity-Jesi 2018 described above, with the exception of the learning rate α being set to 0.1 instead of the original 0.01.

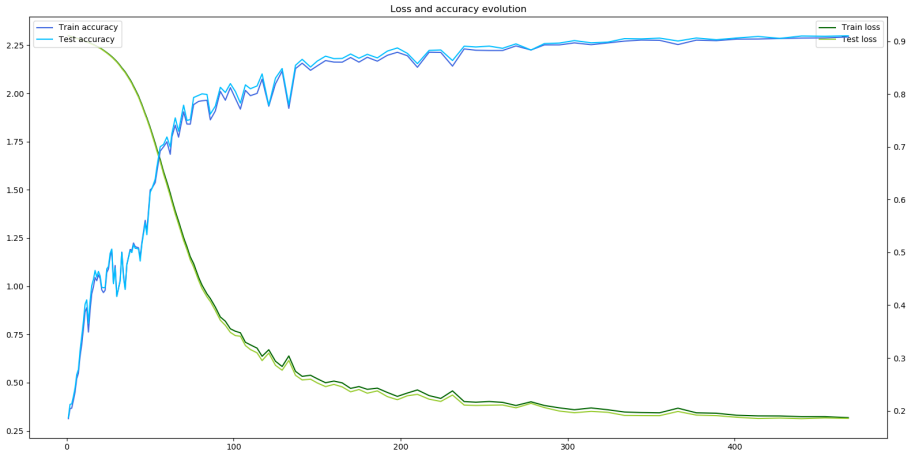


Figure 8: A typical path of the outer SGD, utilising learning rate $\alpha = 0.1$.

2.3.3 Inner SGD - validity of the results

As described in subsection 2.3.1, after a batch of outer SGD is being evaluated and the weights are updated to minimise the loss, inner SGD is started, from the new updated weights, with the aim of minimising G, which is formally defined as:

$$G(\theta) = \frac{1}{2}\|\nabla L(\theta)\|^2 \tag{2}$$

Where θ is the vector of all weights, and $\nabla L(\theta)$ is the vector of the loss gradients at all weights. By minimising G, we are searching for CPs, as it is clear that all loss gradients must be zero when G is zero, which is the definition of the critical point or a trap in the context of DNN.

In order to find the minimum G, the inner SGD search is performed, which differs from the outer SGD only by the fact that it minimises G, not the loss. After implementing this minor change, the SGD minimises G, as presented on Figure 9, using otherwise exactly the same hyper-parameters as outer SGD.

A keen observer may notice that the G is substantially higher than zero, despite us trying to find this minimum. It is to be expected that by using numerical and stochastic methods the exact zero G will not be found, but the G values found should at least be in the range of 1e-10, as recommended in the aforementioned Frye's 2019 work [13]. It is, however, important to remember that a much smaller model was used in Frye's work, and hence the minimum G achievable by SGD in larger models is likely to be larger. Additionally, the actual minimum G attained can be further improved by changing the learning rate and the number of epochs evaluated by the inner SGD; Figure 9 simply demonstrates that the evolution of G under inner SGD exhibits the behaviour expected from a correctly built SGD algorithm, even if not yet calibrated optimally.

The behaviour of the inner SGD presented on Figure 9, however, is not universal. During the later phases of the training process, the inner SGD fails at finding the minimum G, as presented on Figure 10. There is, in fact, a stark transition in the effectiveness of the SGD in finding the minimum G, which occurs roughly at the end of the first regime, as defined in Baity-Jesi [12]. This phenomenon is presented clearly on Figure 11. On this graph, we can see that there is a clear transition in the effectiveness of the inner SGD, around 100th batch of outer SGD, where the inner SGD starts to systematically fail in minimising G, by finding G which is larger than the starting one. This behaviour makes clear that in order for the inner SGD to be efficient at finding minimum G's, the inner SGDs need to be calibrated according to the regime they are operating in.

The empirical experimentation with different learning rates and epochs numbers indicated that low learning rates perform well in the later stages of the simulation, whereas high learning rates are the best for the initial regimes, as presented on Figure 12. From this exercise, I have decided to use learning rate of 0.1 for inner SGDs occurring in the first regime and the learning rate of 0.01 in the second regime, as defined by Baity-Jesi 2018 [12], with 100 epochs evaluated by each inner SGD.

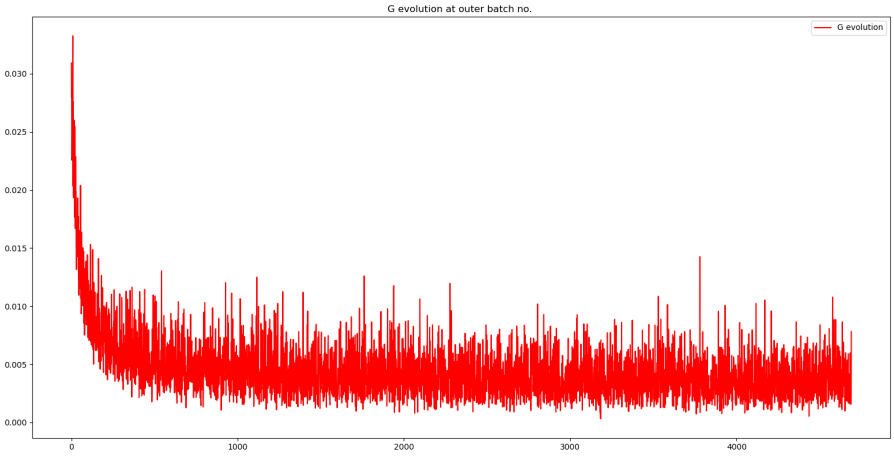


Figure 9: An example single run of the inner SGD, which was calculated for the first batch of the outer SGD. The resultant G is presented on Y-axis and is calculated based on the batch data, not the whole training data, which results in the visible variance of G at each evaluated batch. The evaluated batch number is presented on X axis. The algorithm clearly minimises G quickly, with only random noise around the minimum affecting it for the most of the SGD search time

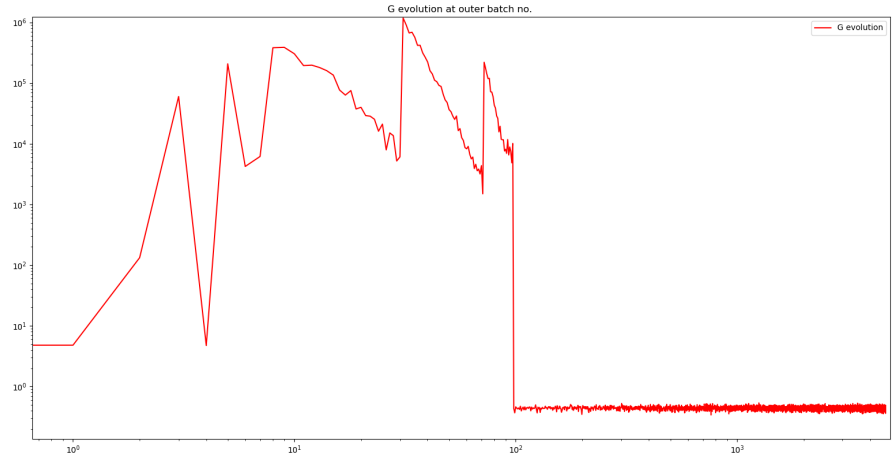


Figure 10: An example of a failed single run of the inner SGD, which was calculated for the 74th batch of the outer SGD. After initial counter-intuitive increases, G suddenly fall and stabilises, but that is due to the exceeding numerical limits of the data structures used to store weights. Please notice the usage of log scales on both axis, necessary to show the scale and the pace of changes in G.

2.4 Results

2.4.1 Defining traps and their similarity - problem description

In this section, I will present the effects of the searches for the critical points, or traps, along the outer SGD. In the context of the trapping time, we are in particular interested in how many points along the outer SGD will converge to the same trap after following an inner SGD, or in the language of GSs, what is the trapping time of each trap.

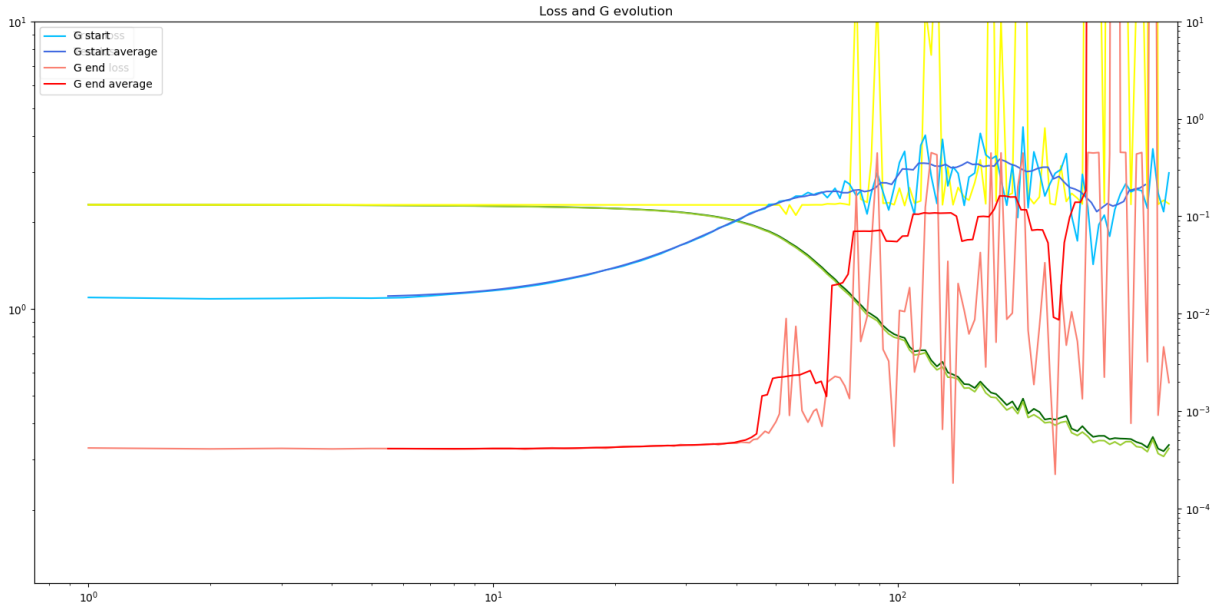


Figure 11: Performance of inner SGD as a function of the Regime of outer SGD. On the left Y axis, the usual error function, with green lines representing the previously defined error and the yellow line represents the training error at the minimum G. On the right Y axis, the G value at the start of the inner SGD (blue line) and at the end of inner SGD (red line), after evaluating one epoch, batches of 128 pictures and learning rate α of 0.1. Whenever the red line is below the blue line, it indicates that the inner SGD has not failed and managed to at least reduce G, which is the case in the first regime, when very little improvement in error along the outer SGD occurs. As the outer SGD progressed to the second regime, of rapid loss improvement, the inner SGD starts to fail as the finishing G is larger than the starting one, or in other words the red line is above the blue one. This sharp transition in the effectiveness of the inner SGD necessitates adjustments of the inner SGD hyper-parameters depending on the regime it should perform in.

Once the trapping times are established, their distributions can be empirically established and evaluated using the GSs tools.

Before the distributions are established, however, a consistent way of determining if the inner SGD converged to the same trap needs to be defined. For example, if I analysed the final minimum G's obtained by two inner SGD, starting from two consecutive batches of outer SGD, e.g. 60th and 61st batch, how can we know that they both reach the same trap? To answer this question, I present 3 methods which, especially when combined, give a robust method of identifying common traps.

2.4.2 Defining traps and their similarity - 1. Loss similarity

We can use analogous definition of a trap to the one from glassy systems- when the loss (which is the equivalent of energy in GS) is the same between the final results of inner SGDs starting from different points of outer SGD. On a diagram, if a set of consecutive outer SGD batches converge to the same trap, the loss obtained from the points which minimise G should be the same, and hence produce a flat line. This is very similar to the graphical representation of the traps presented on Figure 3, where flat areas indicate the existence of the trap and their length are the trapping time. As the outer SGD progresses, and batches start to converge to a different trap, we would expect the loss at the minimum G to change sharply and stay at the new level, resulting in a step-wise curve on a diagram.

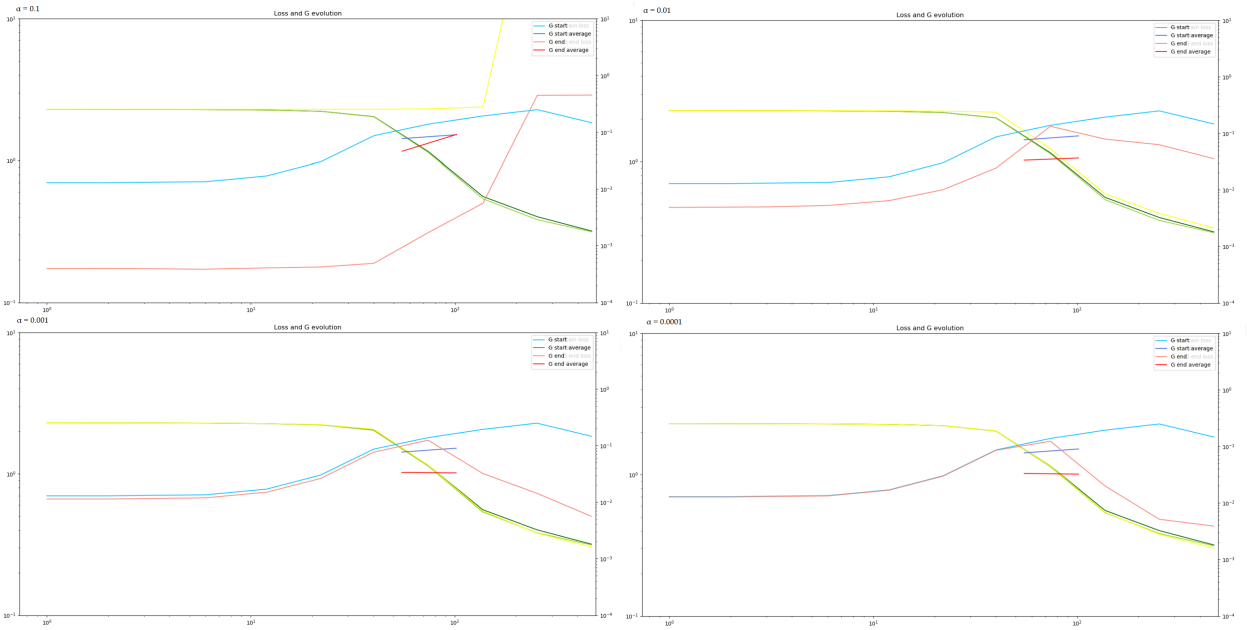


Figure 12: Performance of inner SGD with different learning rate α, evaluated over 10 epochs and at 10 points of the outer SGD.

2.4.3 Defining traps and their similarity - 2. G similarity

Another indicator of reaching the same trap would be the stability of the final G across consecutive inner SGD. In theory, given differentiability of the loss landscape, it should be always in principle possible to reach exact zero gradient or zero G. In practice, however, due to using approximate, numerical methods it may well be the case that the minimum G reached in a trap is a unique constant, and different from the minimum G attained in other traps. Therefore, as the traps attained are changing, the curve of the minimum G is likely to exhibit similar step-wise behaviour, like the loss curve, which would also allow us identify trapping time.

2.4.4 Defining traps and their similarity - 3. weights similarity

The final and perhaps the most obvious way of defining the same trap is by comparing the exact point it occupies on the loss landscape, or in other words by comparing the weights of the network which result in the minimum G obtained by the inner SGD. An intuitive way of defining the Weights Difference (WD) between two points A and B would be to take the difference of the two weights vectors at points A and B and calculate the norm of the resultant vector:

$$WD(A, B) = \|\theta_A - \theta_B\| \tag{3}$$

Choosing different points as A and B could provide us with different insights. For example, if a two consecutive inner SGDs converge to the same trap, we would expect the WD of the two minimum Gs to be zero. Nevertheless, due to discontinuous behaviour of the DNN (ReLU activation functions) it is possible to obtain exactly the same loss and G from a hyper-space of weights, which means that expecting the results to converge to exactly the same weights within a trap is unreasonable. It is, however, true that the consecutive points producing minimum G which land in the same trap should be relatively "close" to each other in terms of their weights, that is have "small" WD. If, on the other hand, the inner SGD from the next evaluated batch converges to a different trap, the recorded WD between the minimised Gs from those two batches should "jump", and fall back to the "small" level for the next batch, as now the two consecutive inner SGDs converge to the same trap. On a diagram therefore, we should see distinct "peaks" of WD between consecutive ends of inner SGDs, which will be indicating that a new trap is being reached. This WD comparison will be called "end-to-end WD", as it compares the end results of two consecutive inner SGDs.

In terms of defining what should be considered as "small" WD, we can compare it to the WD of the consecutive batches of the outer SGD, which are effectively WD of the starting points of the inner SGDs. As the starting

points of the inner SGDs are moving, it is reasonable to expect that the ending points of inner SGDs, where G is minimised, would also move to some extent. If the WD between the consecutive ending points of SGDs is similar or, better still, smaller than between the starting points, it does suggest that the same trap has been reached, as the weights difference is small enough. The WD between the batches of the outer SGD can therefore be seen as a "background noise" or a "yardstick" of how large WD we can in general expect in the dynamics. This WD comparison will be called "out SGD WD" as it compared the consecutive batches of the outer SGD.

Alternatively, instead of looking at the WD between end points of consecutive inner SGDs, we can calculate WD between the starting and ending point within each inner SGD. This would be a measurement of how far the inner SGD "travelled" across the loss landscape to find the minimum G. If a group of inner SGDs are converging to the same minimum G trap, then they should all have similar distance travelled to this trap, with the small differences accounted for the movement of the starting points. Convergence to a different trap should be associated with a sharp change in this distance travelled. Finally, this WD comparison will be called "start-to-end WD", as it compares the start, or beginning of a particular inner SGD to its final, ending result.

In the results presented below, we will analyse all of these definitions of traps to investigate the trapping behaviour. The results will be divided into two sections. The results will be evaluated separately for the first and second regime, as defined by Baity-Jesi 2018 [12].

2.4.5 1st Regime - individual results

As there is no clear-cut definition of how long each learning regime lasts, in the context of this analysis the first regime has been defined as the first 50 batches evaluated by the outer SGD. This boundary seems to be a reasonable choice by inspecting Figure 8, but it also will be further strengthened when comparing fundamental trapping behaviours. The results from the first phase analysis are plotted with the whole one epoch of the outer SGD in Figure 13, for the sake of visualising where the analysis is performed.

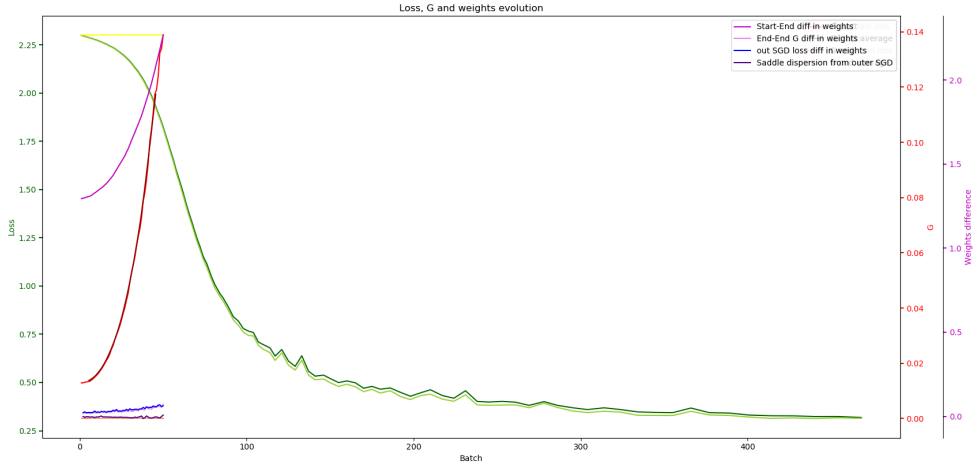


Figure 13: The results of the first phase analysis with the whole outer SGD for comparison.

Due to the results being focused in a small part of the graph, as well as large differences between the readings, Figure 14 present a more useful view, focused on the Phase 1 analysis results and employing log scales. As defined in the beginning of the Results section, we will now evaluate these results by looking at three definitions of trapping points similarity.

2.4.6 1st Regime - loss similarity

All loss results are plotted on the left Y-axis. Dark and bright green curves represent the training and testing loss of the usual outer SGD, similarly to the colour coding of Baity-Jesi 2018 [12]. Because the inner SGD start from the outer SGD, these green lines can also be though of as the starting loss of the inner SGD. Yellow line, as described on Figure 11, indicates the training loss at the end of the inner SGD, i.e. the minimum G obtained. One can see that despite the (clearly expected from the outer SGD minimising the loss) decreases in the loss, inner SGDs

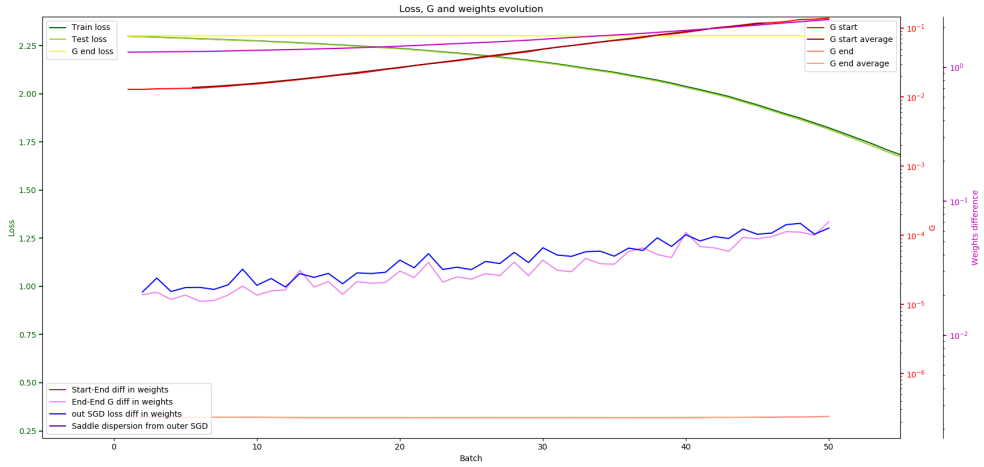


Figure 14: The results of the first phase analysis. Please refer to Sections 2.4.6, 2.4.7 and 2.4.8 for the explanation of the presented behaviour.

always converge to the same loss value. This suggests that there is one trap to which inner SGDs in the first phase converge.

2.4.7 1st Regime - G similarity

A very similar story is found emerges from analysing the changes in G. G results are presented on the first RHS Y-curve. The bright red curve indicates the starting G, i.e. the G at the beginning of the inner SGD or directly on the outer SGD. The salmon colour curve is the minimised G, at the end of the inner SGD. Despite the starting G slowly rising, the end G stays constant across the simulation and way below the starting G, indicating a good performance of the inner SGD as well as the convergence of all inner SGDs to a single trap.

2.4.8 1st Regime - weights similarity

Evolution of WDs, which are plotted on the second RHS Y-axis also support the view that a single trap is attracting all inner SGDs. The end-to-end WDs , marked in pink, are showing no visible spikes, which would be a sign of trap changes. Additionally, in almost all batches the end-to-end WD is below the out-SGD WD, or WD of consecutive batches of the outer SGD, marked in blue, and is strongly correlated with it. This further supports the view that inner SGDs converge to the same trap as the end points of the consecutive inner SGDs are "closer" to each other than the starting points, and any variation in the distance between the end points is mostly explained by the variation of the starting points.

In terms of the start-to-end WD, i.e. the WD between the starting and the end point of within each inner SGD, shown as the dark purple curve, it is growing at an increasing rate. This continuous increase of WD between start and end of inner SGDs can be potentially explained by the inner SGDs having to "travel back" further and further back to the same trap as the loss minimisation (represented here by the blue line) pushes the starting point further away from the trap. More importantly, however, there seem to be no discrete jumps in start-to-end distance, which would indicate trap convergence change.

The above analysis was repeated 3 times, using different initial weights and random numbers and it always produced the same result- constant loss and G at the end of the inner SGD, no spikes in end-to-end WD and continuous increases in start-to-end WD. Therefore, from the above analysis, I conclude that there is only one trap to which the dynamics converges to in the first learning regime. Hence, it is not possible to describe any time trapping distributions.

Although not particularly noteworthy behaviour was identified in the first regime of the dynamics, the situation becomes more interesting in the second phase, which will be presented in the next sections.

2.4.9 2nd Regime - individual results

As with the first regime duration, the second regime was empirically defined to start from 51st batch and end on the 150th batch. An example of the analysis, with the whole outer SGD, is presented on Figure 15.

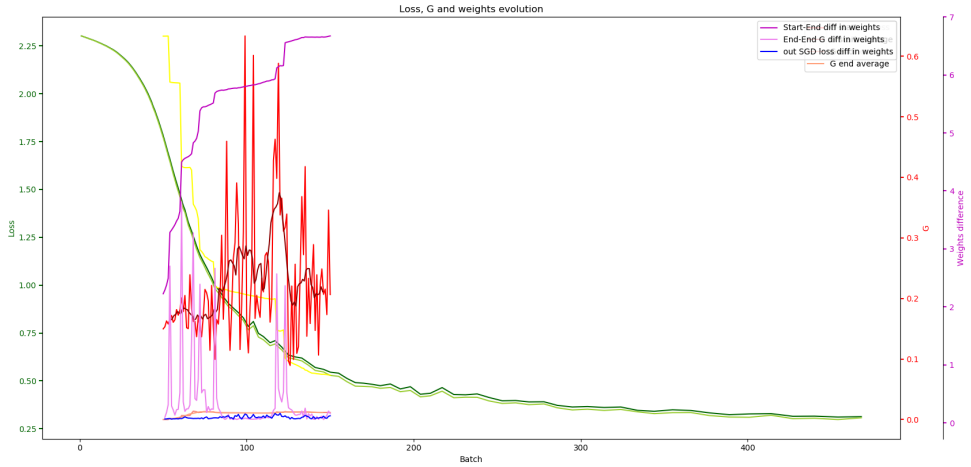


Figure 15: The results of the second phase analysis with the whole outer SGD for comparison.

As with the first phase analysis, the more legible results are presented on Figure 16. The in-depth analysis of the presented results is provided in the following sections.

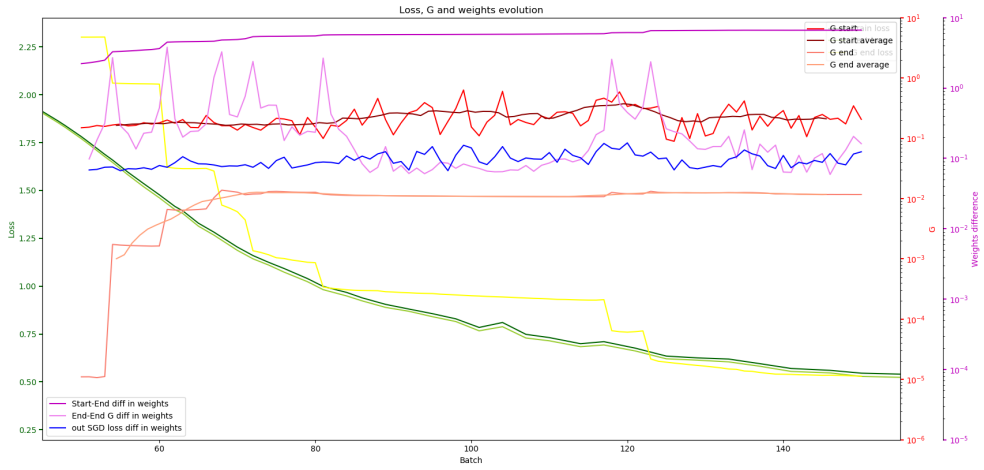


Figure 16: The results of the second phase analysis. Please refer to Sections 2.4.10, 2.4.11 and 2.4.12 for the explanation of the presented behaviour.

2.4.10 2nd Regime - loss similarity

Contrary to the loss at the end of the inner SGDs in the first regime, where there was no change in the yellow line, here we observe a clear- step-wise changes in the loss at the minimum G. This indicates that the inner SGDs do indeed start to converge to different traps as the outer SGD advances. The length of the "ledges" of the yellow curve would then define the trapping time in each of the traps, which can be measured and the distributions can be compared within different parts of the second regime. Just by looking at this individual result, it does seem that the average trapping time increases in the later parts of the analysis.

2.4.11 2nd Regime - G similarity

Evolution of the minimum G, presented by the salmon curve, again provides a more complex behaviour when compared to the first regime. We observe step-wise changes in the G attained, which coincide with the steps of the yellow loss curve, further supporting the view that distinct traps are being reached from different points along the outer SGD. Even in the later stages, where the salmon curve is almost flat, step-wise behaviour can still be observed when zooming in on the end G curve, as presented on Figure 17

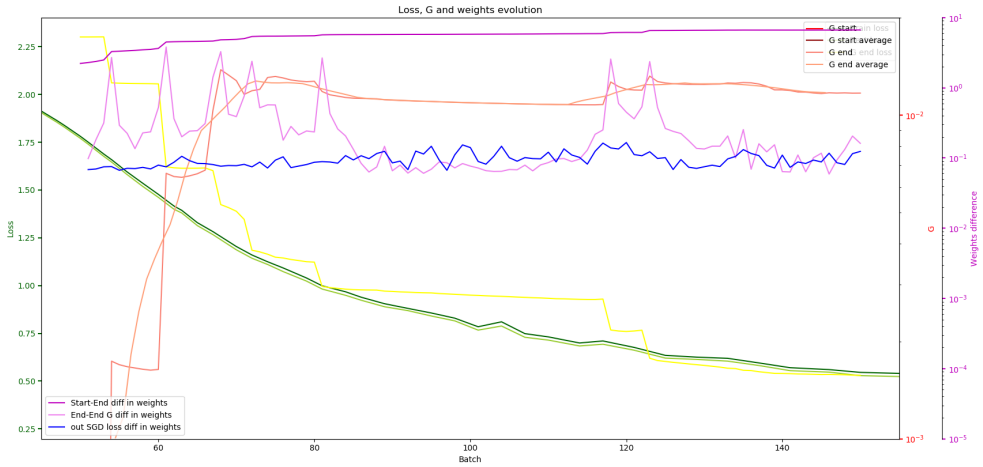


Figure 17: The results of the second phase analysis with G scaled to emphasise the step-wise behaviour in the later stages of the analysis.

What is, however, unusual in the behaviour of the minimum Gs obtained. It is unusual that they seem to be increasing, even if in the step-wise fashion, pushing them substantially further away from the low levels achieved in the earlier inner SGDs, not to mention substantially further away from the G levels to be obtained as indicated by Frye 2019 [13].

It might be the case that this unusual behaviour simply indicates that, although there exists a lower minimum G, it is not attained by the inner SGD, which is a local method. This outcome therefore could be a feature of the SGD algorithm which gets stuck in the closest local minimum of G.

Alternatively, it could be the case that, even though there is a better minimum G which could have been found by the SGD, it is not obtained because the learning rate was not small enough. This would mean that the reported G is not the minimum, but even if the minimum G was not attained, the inner SGD at least moved to the area of attraction of the trap, as indicated by the other two trap indicators. With this caveat in mind, I would argue that end G curve indicates at least the basins of attraction of different traps, which is sufficient from the point of view of analysing trapping time.

2.4.12 2nd Regime - weights similarity

What is, in my opinion, the most interesting finding of this analysis is the clear emergence of the peaks in the pink end-to-end WD curve. As predicted, there are "jumps" in the distance between consecutive end points of inner SGD, or minimum Gs whenever they are being attracted to different trap. These jumps coincide exactly with the steps of the yellow loss function, or salmon minimum G function, further supporting the emergence of clear traps with distinct trapping time. The apparent steps in the violet start-to-end WD curve provide more evidence for the existence of separate traps- inner SGD suddenly have to "travel" more to find the new minimum G.

Although the peaks in end-to-end WD do support the view of separate traps, there is still substantial WD in between the peaks. For the later traps the end-to-end WD seem to get close and even below our yardstick variation of the out-SGD WD (the blue line), but that is not the case for the early traps, where we would expect the end-to-end WD to be lower between the peaks. I suspect that this could be caused by the same reason minimum G is increasing- failure of the inner SGD to find the actual minimum G, with the final weights "bouncing" in the basin of attraction of the trap, but never reaching it and hence creating relatively high end-to-end WD. Nevertheless, the

emergent clear peaks are a strong indicator that there is a drastic change in the weights obtained between traps, but with further adjustments of the epochs evaluated in learning rate the "valleys" in the pink curve may be even deeper and more pronounced.

The above example results show clear changes in the traps to which the dynamics converge. The fact that the similar trapping behaviour can be obtained in DNN setting is a key finding of this analysis. As it is possible to calculate trapping time, it is within our reach to calculate trapping distributions as well as their distributions, in a fashion similar to GS setting. A first attempt at analysing these trapping times will be performed in the next section.

2.4.13 2nd Regime - trapping time analysis

The second regime analysis was conducted 9 times using different randomly generated starting weights as well as random numbers. Due to a single analysis taking close to two days to finish, it was not possible to run more analyses for the purpose of this thesis.

The length of a particular trapping time was calculated as the difference between the two peaks of the end-to-end WD difference curve. As the X-axis indicates the batch of the outer SGD, the trapping time is therefore measured in terms of batches of outer SGD which, after following the inner SGD, converge in the same trap. Although unnecessary due to their agreement, the other two definitions of traps (the ledges of the loss or G function) were also considered in deciding on the length of the trapping time.

Using the results from the whole range of second regimen (batches 51 - 150), 55 trapping times were obtained: the average trapping time was calculated to be 12.04, with variance of 116.36. The actual experimental distribution is presented on Figure 18.

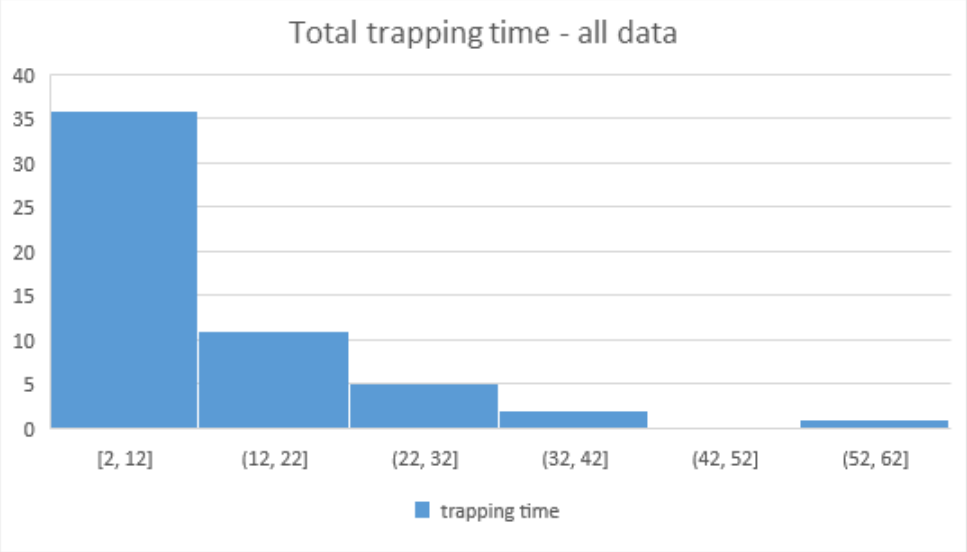


Figure 18: Histogram depicting the trapping time obtained from 9 analyses performed. Total number of trapping times in each group depicted on the Y axis

The large variance in comparison to the mean could indicate that the actual variance is infinite, potentially suggesting a power law guiding the distribution of trapping time. To further test that, the frequencies were plotted on the log scale and are presented on Figure 19, with the highest group with a single reading omitted. Despite being based only on 54 readings, the graphs shows relatively straight line for the first 3 groups, with the linearity breaking only with the 4th group which consists of only two readings. With such a small sample size it is difficult to make any strong conclusion, but the graph provides some tentative support of power law guiding the trapping time distribution, which could be further investigated.

It is also interesting to understand how the trapping time distributions change in different parts of the simulations. Due to relatively few points to analyse and in order to keep a reasonable amount of data in both groups, the results we split into two groups: the traps which started before or on the 80th batch of the outer SGD (which will be called "early" group) and the traps which started after 80th batch (which will be called "late" group), which resulted in 35 and 20 traps in the two groups respectively. The early group had the average trapping time of 7.94

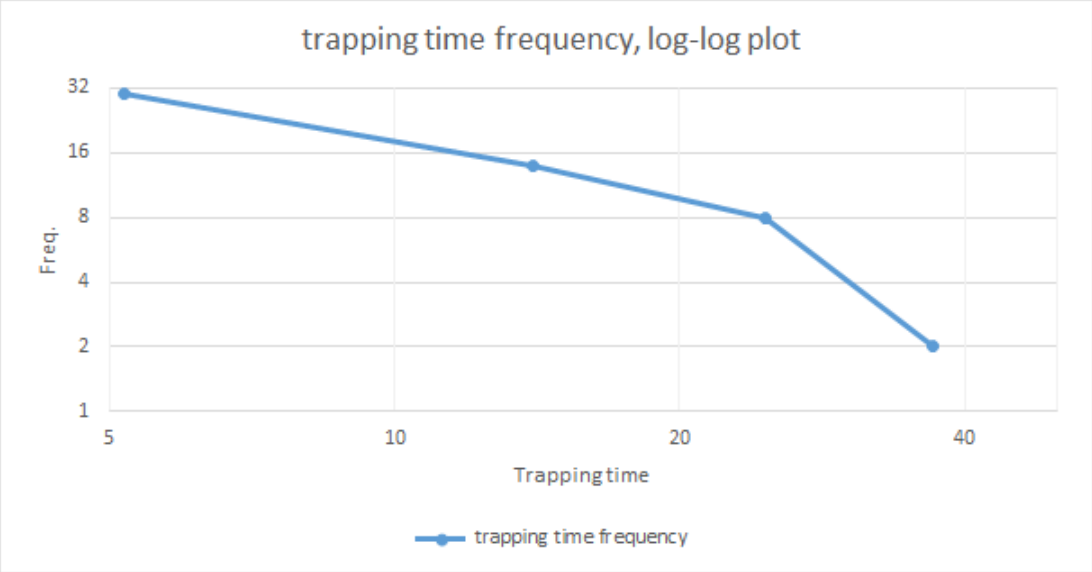


Figure 19: Logarithm of the trapping frequencies obtained experimentally. For the x-axis, the average of the trapping times of the group was used to identify a single x-coordinate

with variance of 30.74. The late group had average trapping time of 19.2 and the variance of 185.56. Both of these distributions are presented on Figure 20.

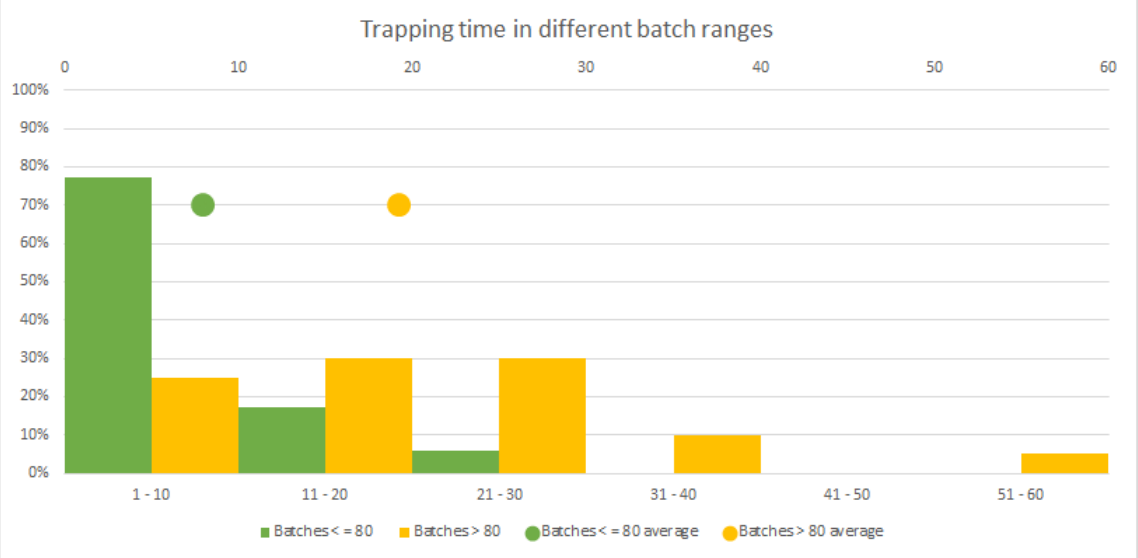


Figure 20: Histogram depicting the trapping time split into two groups, early and late. The proportion of trapping times in each group depicted on the Y axis, with average values shown as large dots, plotted on the top X-axis for the comparison (Y-axis value is arbitrary for the averages).

As it was identified earlier, in the later phases of the second regimen, the trapping time increases (more than doubles in our simple comparison). This increase in trapping time could be connected to the slowing of the dynamics, as the pace at which the loss is minimised decreases as the simulation progresses.

In essence, the above analysis shows that the trapping times in deep neural networks can be obtained and used in the analysis similar to the one performed in the models of glassy systems.

2.5 Discussion and Conclusion

This thesis attempted at creating an additional connection between deep neural networks and glassy systems. It started by describing the similarities of the DNNs and GS and pointed out that the analysis of DNN could benefit from further utilisation of tools used for the analysis of glassy systems. It then proceeded to outlining the work already performed in the field by the researchers. With this knowledge, it set up a system which would allow to create the result, trapping time, in the DNN setting which would open the way of using more advanced techniques employed in the analysis of GSs. Although this was not possible in the first learning regime, by combining three different definitions of what traps are, it showed that it is indeed possible to obtain similar trapping times in DNNs' second learning regime, opening this path to further analysis. It also produced a preliminary analysis of trapping times itself, showing that the average trapping time is increasing with the number of batches evaluated. It has also shown that the distribution of the trapping time could be guided by a power law, which would support the view expressed by Baity-Jessi of the second regime exhibiting aging, or "glassy" behaviour, but this finding should be confirmed with larger sample size [12].

There is a number of paths through which these results could be further utilised. The third learning regiment, where slow improvement in the loss occur, where the dynamics was attributed to diffusion by Baity-Jesi, could be evaluated for the trapping time occurring in this regime [12]. Indeed, as shown in this thesis, different regimes identified by Baity-Jesi have distinct behaviours trapping behaviours and the trapping behaviour could be used to formally define different learning regimes [12].

Furthermore, this thesis focused on an over-parametrised network, which is capable of reaching approximately zero loss levels. Under-parametrised networks can be analysed in a similar way, and the preliminary results conducted indicate that their trapping behaviour differs from over-parametrised networks: with traps being longer in terms of trapping time as well as having no apparent order in terms of time evolution. Figure 21 presents two results which were evaluated for under-parametrised network, with 5-5-10 structure, as opposed to 100-100-10 used in the main analysis. Please notice that the analysis was shifted to 100-200 batches as it was experimentally established that the transition to the second phase occurs later.

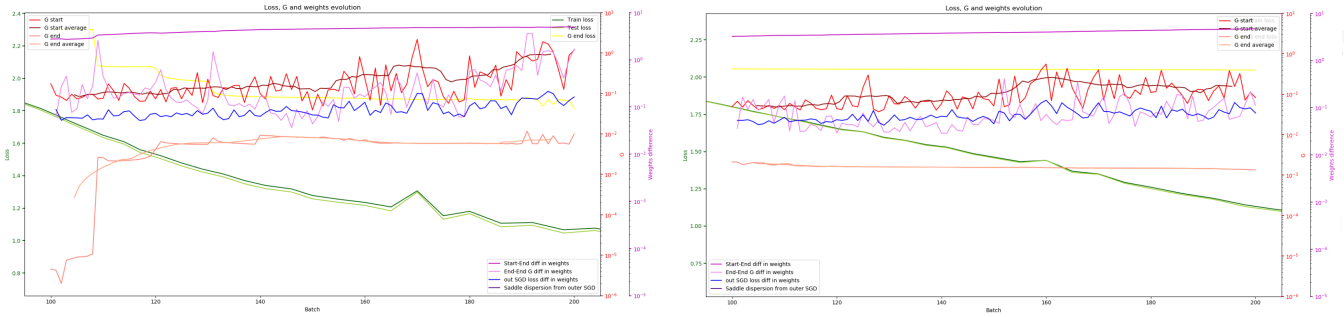


Figure 21: Two examples of simulations on 5-5-10 network.

A number of facets of this analysis could be improved to further strengthen the conclusions of this thesis. First, it should be investigated if lowering the learning rate in inner SGDs as the simulation progresses allows the inner SGDs to reach lower G levels, especially in the later phases of the second regime, where minimum G increases. Hopefully this would improve the issue of relatively large end-to-end WD between the "peaks" indicating trap changes. Alternatively, analysing more epochs in the inner SGDs could also be investigated, which could not be performed here due to computational constrains. Finally, with only 55 trapping times available it is difficult to draw any strong conclusions about the behaviour of the trapping time.

Nevertheless, the main aim of this thesis was to show that trapping time distributions along outer SGDs are well defined and can be calculated to perform analysis with the tools developed for GSs, and this aim has been achieved. Further studies would be required to full utilise this potential new path of analysing Deep Neural Networks.

References

[1] M Alba, J Hammann, M Ocio, Ph Refregier, and H Bouchiat. Spin-glass dynamics from magnetic noise, relaxation, and susceptibility measurements. *Journal of Applied Physics*, 61(8):3683–3688, 1987.

[2] Eric Vincent. Ageing, rejuvenation and memory: the example of spin-glasses. In *Ageing and the glass transition*, pages 7–60. Springer, 2007.

[3] Maira Pavlovskaia, Kewei Tu, and Song-Chun Zhu. Mapping the energy landscape of non-convex optimization problems. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 421–435. Springer, 2015.

[4] J-P Bouchaud and David S Dean. Aging on parisi's tree. *Journal de Physique I*, 5(3):265–286, 1995.

[5] Alain Barrat and Marc Mézard. Phase space diffusion and low temperature aging. *Journal de Physique I*, 5(8):941–947, 1995.

[6] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[7] Donald F Specht. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[8] Viktor Dotsenko. *An introduction to the theory of spin glasses and neural networks*, volume 54. World Scientific, 1995.

[9] Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Spin-glass models of neural networks. *Physical Review A*, 32(2):1007, 1985.

[10] Marco Baity-Jesi, Giulio Biroli, and Chiara Cammarota. Activated aging dynamics and effective trap model description in the random energy model. *Journal of Statistical Mechanics: Theory and Experiment*, 2018(1):013301, 2018.

[11] Chiara Cammarota and Enzo Marinari. Spontaneous energy-barrier formation in entropy-driven glassy dynamics. *Physical Review E*, 92(1):010301, 2015.

[12] Marco Baity-Jesi, Levent Sagun, Mario Geiger, Stefano Spigler, G Ben Arous, Chiara Cammarota, Yann LeCun, Matthieu Wyart, and Giulio Biroli. Comparing dynamics: Deep neural networks versus glassy systems. *arXiv preprint arXiv:1803.06969*, 2018.

[13] Charles G Frye, Neha S Wadia, Michael R DeWeese, and Kristofer E Bouchard. Numerically recovering the critical points of a deep linear autoencoder. *arXiv preprint arXiv:1901.10603*, 2019.

[14] Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.