Analysing Deep Neural Networks as Brain Models

Conor Mullaney

14th December 2022

1 The Biological Relevance of Back propagation

1.1 How does the algorithm work? (494/500)

Backpropagation is an algorithm used to train neural networks. It calculates the gradients between an output of the network and a target vector via a loss function. This error is passed back through the network through internal hidden units, implementing weight adjustments to facilitate learning [1].

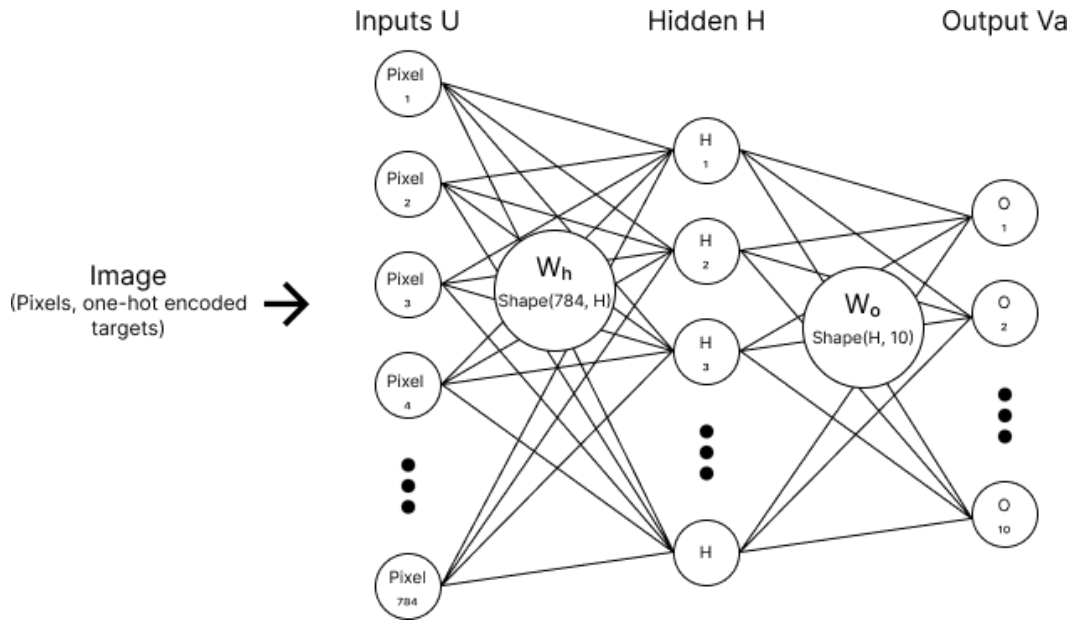


Figure 1: *Neural network architecture for the fashion MNIST data-set [2]*

The Algorithm:

The variables used in the equation, relate to those displayed in Figure 1.

1. Forward Pass: Backpropagation begins with a forward pass on a set of inputs. These inputs are mapped between layers through weight matrices. The steps of the forward pass are as follows:

   (a) Hidden Layer: $Z = UW_h$

(b) Activation function $f(x)$ applied to the hidden layer: $H = f(Z)$

(c) Output layer $V = HW_o$

(d) Activation function f (x) applied to the output layer: $V_a = f(V)$

2. Calculating Loss: A typical MSE loss function is applied to compare a target output T with the output of the network Va with batch size n.

$$MSE = \frac{1}{2}\sum_{i=1}^{n}(Va_i - T_i)^2$$

$$\frac{\partial \mathscr{L}}{\partial Va} = \sum_{i=1}^{n}(Va_i - T_i)$$

3. Backpropagation: We update our weight matrices by calculating the gradient of the loss function with respect to the weights at each layer. Figure 4b shows the change in magnitude of each component of the gradients over learning. The figure suggests most of the learning is achieved through adapting W_o. The implementation of this is represented in Figure 2, with the following equations:

(a) Calculating derivatives

$$\frac{\partial \mathscr{L}}{\partial V} = f'(V) \odot \frac{\partial \mathscr{L}}{\partial V_a}$$

$$\frac{\partial \mathscr{L}}{\partial W_o} = H^\top \frac{\partial \mathscr{L}}{\partial V}$$

$$\frac{\partial \mathscr{L}}{\partial H} = \frac{\partial \mathscr{L}}{\partial V} W_o^\top$$

$$\frac{\partial \mathscr{L}}{\partial Z} = f'(Z) \odot \frac{\partial \mathscr{L}}{\partial H}$$

$$\frac{\partial \mathscr{L}}{\partial W_h} = U^\top \frac{\partial \mathscr{L}}{\partial Z}$$

(b) Weight Updates: Update weights with a learning rate lr:

$$W_o = W_o - (r \odot \frac{\partial \mathscr{L}}{\partial W_o})$$

$$W_h = W_h - (r \odot \frac{\partial \mathscr{L}}{\partial W_h})$$

```python
def backprop(Wh, Wo, dL_dPred, U, H, Va, Z, V):
    dLdV = np.multiply(sigmoid_prime(Va), dL_dPred)
    dL_dWo = np.matmul(H.T,dLdV)
    dL_dH = np.matmul(dLdV, Wo.T)
    dL_dZ = np.multiply(sigmoid_prime(Z), dL_dH)
    dL_dWh = np.matmul(U.T, dL_dZ)
    return dL_dWh, dL_dWo
```

Figure 2: *Backpropagation function of our neural network*

Activation functions allow non-linearity in the network, and we have considered both a sigmoid and tanh activation function for the hidden layer. Although Figure 3 shows tanh performs marginally better than sigmoid, the rest of this work will implement sigmoid. A sigmoid is applied on the output layer to normalise the output and compare with the one-hot encoded target vector T. These functions suffer from the vanishing gradient problem [3] and are a potential reason for the lack of learning in some gradient components, Figure 4b.

$$sigmoid(x) = \frac{1}{1+e^{-x}}$$

$$sigmoid'(x) = sigmoid(x) \cdot (1 - sigmoid(x))$$

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

$$\tanh'(x) = 1 - (\tanh(a) * *2)$$

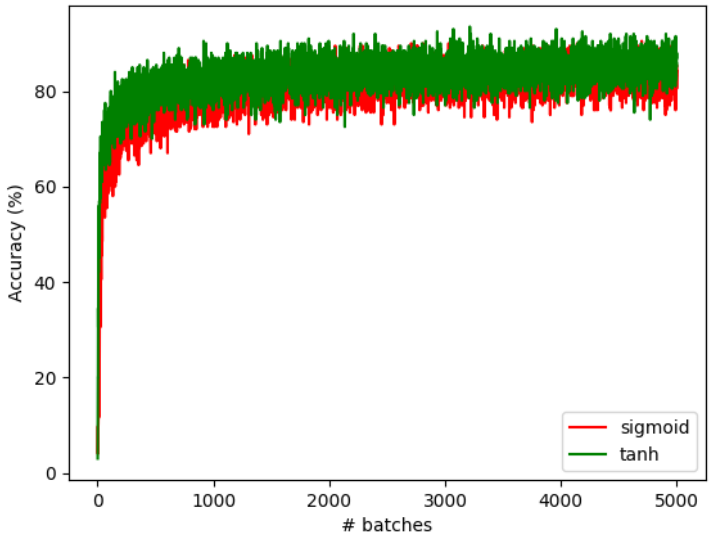
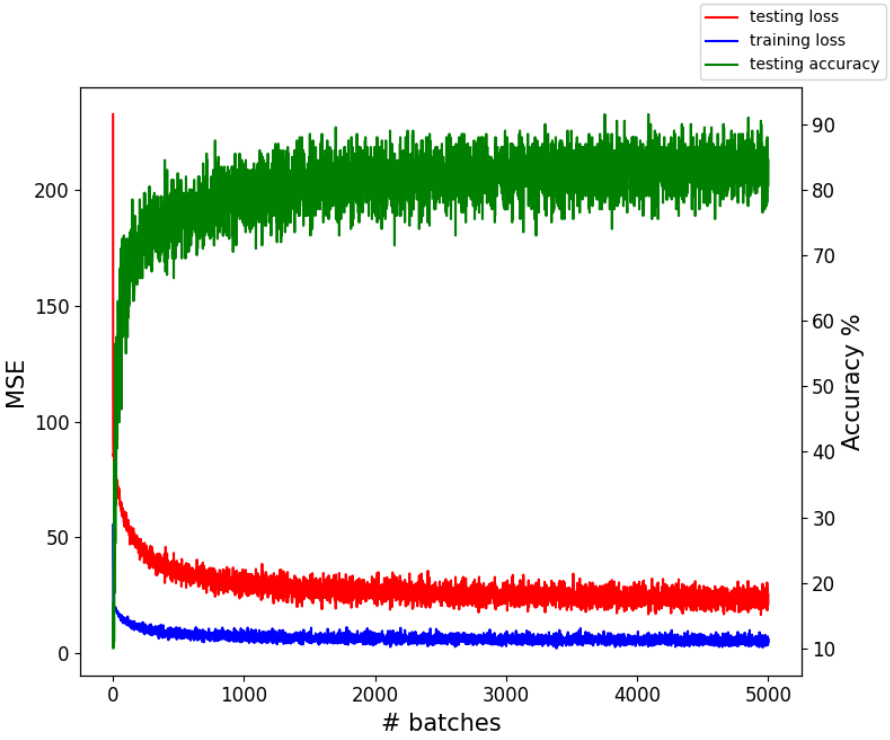
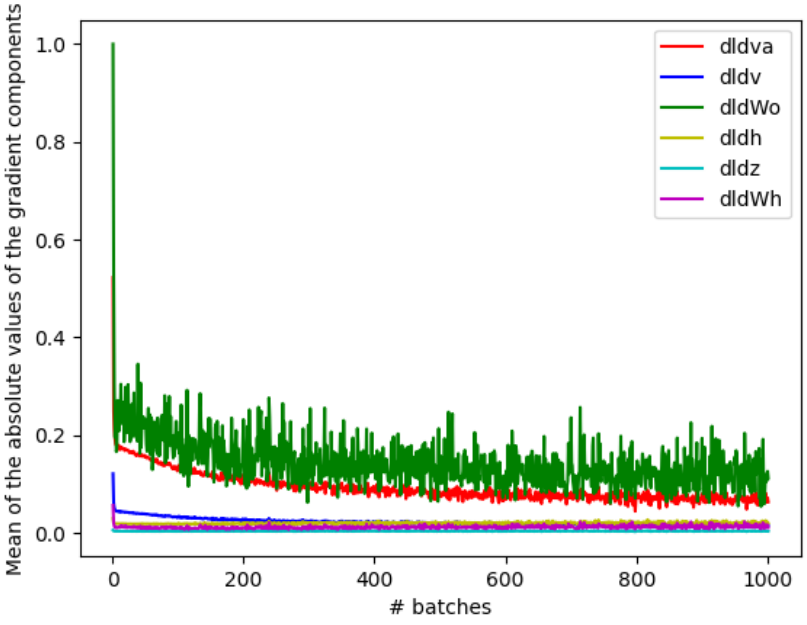


Figure 3: *Different activation functions on the hidden layer: Hidden Size* $H = 112$*, Batchsize* $bs = 48$*, Test Batchsize* $tbs = 200$*, learning rate* $lr = 0.01$



(a) *Losses and accuracies of the network over learning*



(b) *Mean magnitude of the gradient components over learning*

Figure 4: *Backpropagation demonstration of our neural network.* $H = 112$, $bs = 48$, $tbs = 200$, $lr = 0.01$

1.2 How does the algorithm relate to the brain? (484/500)

Error-backpropagation has remained as the most common solution to the credit assignment problem in the brain [4]. However, due to the non-linearity of neurons, traditional backpropagation is considered biologically implausible.

Weight Transport problem

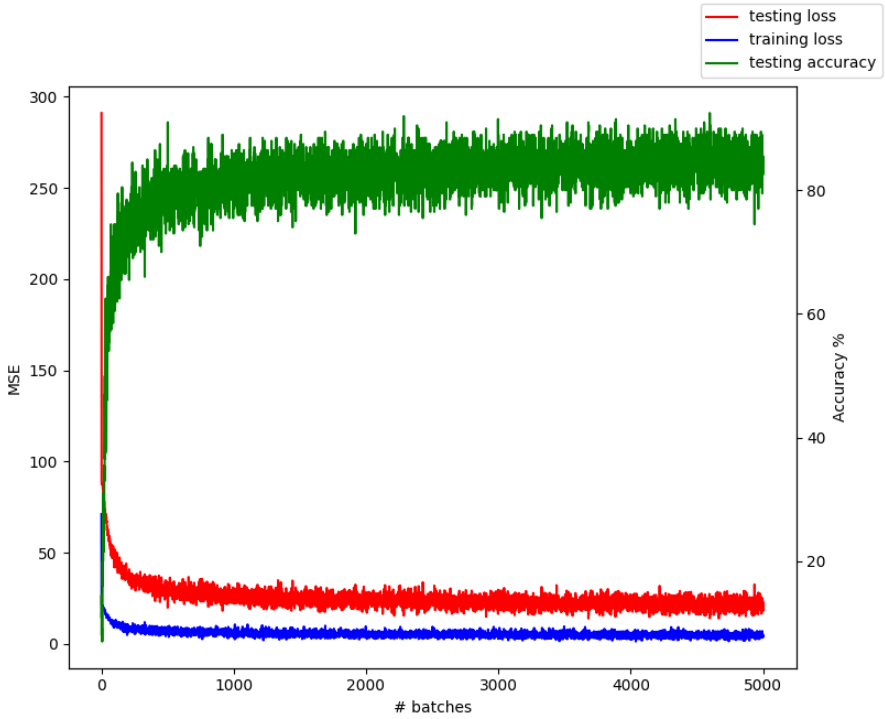
Symmetric feedback and feedforward weights would require precise network connectivity thought to be impossible in the brain [5]. However, static random feedback weights still provide a viable backpropagation learning solution as demonstrated in Figure 5a which shows comparable learning to symmetric weights. This provides biologically plausible feedback weights for backpropagation. Taking this a step further, Figure 5b shows some learning through random weights which change at each iteration. However, the level of accuracy gained from this learning is not significant enough to imply that this implementation of feedback weights is representative of the brain.

Derivative of the activation function

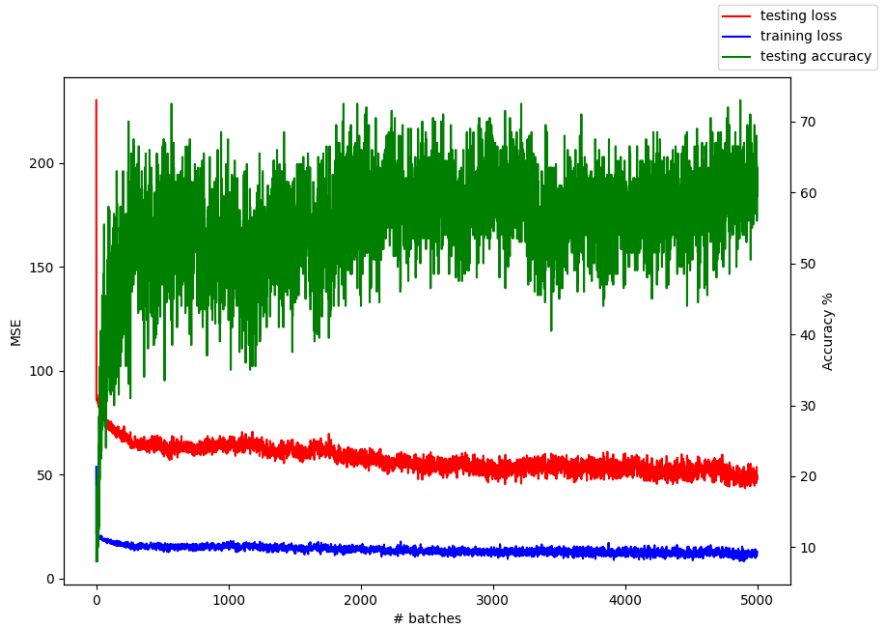
It is unlikely that neurons in the brain would be able to perform mathematical operations like calculating derivatives, as this type of computation is not within their usual scope of function. Therefore, it is biologically implausible for neurons in the brain to calculate their own derivatives. Figure 6 displays the effect of no derivative of the activation function being applied to the backpropagation algorithm. Although Figure 6a initially learns, as learning progresses the accuracy of the system declines. Figure 6b shows this decline primarily present itself in the weight updates of W_h which varies significantly compared to Figure 4b. This suggests that without knowledge of the derivative of the activation function, the gradients cannot be backpropagated properly which results in large fluctuations of the weight W_h.

Two phase learning

Two phase learning presents the most significant barrier to suggesting that the brain can implements traditional backpropagation to update its synapses. This is because it can be argued that there is no separation between the training and learning phase within the brain [6]. However, literature presents mechanisms to implement single-phase learning through the use of continuous local prediction errors [7, 8]. We have tried to replicate the stochastic nature of learning in the brain in our network through randomly propagating our learning to W_h with probability p. Figure 7 shows the network fails to learn in any significant manner as the test losses remain significantly higher than traditional backpropagation shown in Figure 4a, even for high values of p.

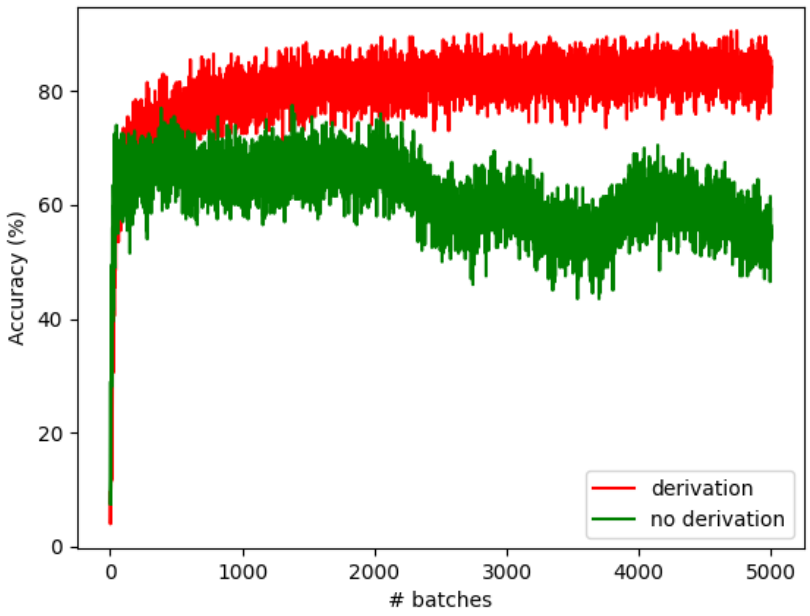


(a) *Fixed random feedback weights*

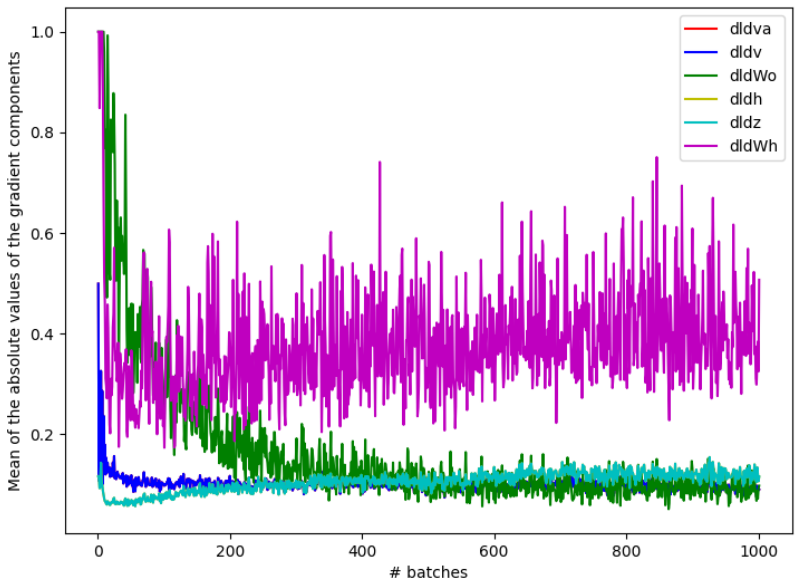


(b) *Random feedback weights for every backpropagation phase*

Figure 5: *Displaying the effect of random feedback weights on the backpropagation algorithm,* $H = 112$*,* $bs = 48$*,* $tbs = 200$*,* $lr = 0.01$



(a) *Effect on accuracy over learning*



(b) *Effect on the components of the gradient function*

Figure 6: *Displaying the effect of no knowledge of the derivation of the activation function on the backpropagation algorithm,* $H = 112$, $bs = 48$, $tbs = 200$, $lr = 0.01$

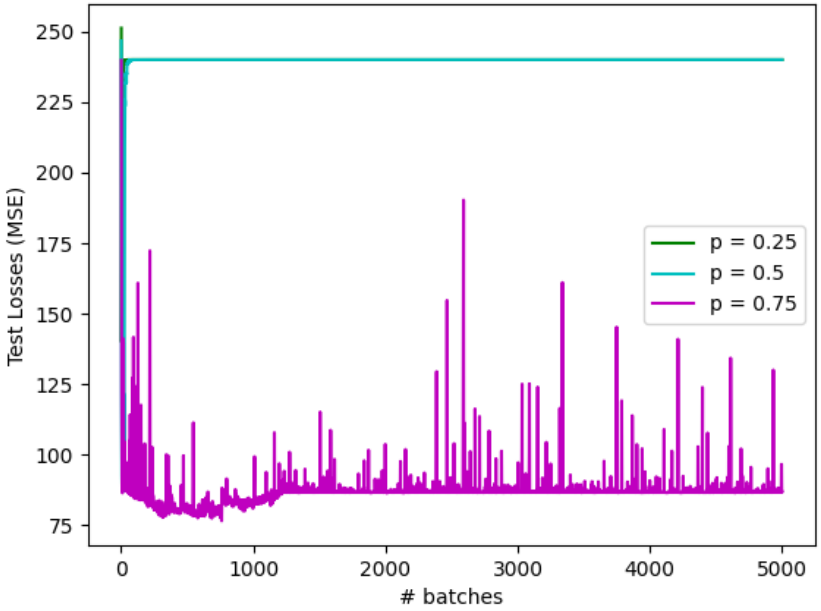


Figure 7: *Test losses of the network over learning given a probability of executing backpropagation in* W_h $H = 48$, $bs = 48$, $tbs = 200$, $lr = 0.01$

1.3 A Comparison of Learning paradigms (491/500

Machine learning aims to use algorithms and statistical models to perform a task which a program is not specifically programmed to do. However, due to the wide variety of tasks and learning environments, there is no one-size fits all algorithm [9]. Supervised learning algorithms are implemented on labelled datasets. This is similar to how the brain learns through past experiences, using past examples to make new predictions about new situations. Unsupervised learning is best suited to data with no "correct" answers. The algorithms discover and present relevant features within the data without external assistance. This learning relates to how the brain learns about it's environment through exploration and interaction. Finally, reinforcement learning examines how agents should make actions in an environment to maximise a notion of reward. This type of learning relates to how the brain learns to make new decisions based on the consequences of its actions. It is unreasonable to state that the brain uses one-type of learning more than another as the interactions of the brain to the environment is a complex task. However there have been varying levels of success in implementing these algorithms with biological plausibility, each being more plausible depending on the specific context in which they are implemented.

Advantages of supervised learning

Supervised learning has a few advantageous over its counterparts in terms of performance. Since the model is trained on a large amount of labelled data and is able to make use of this to improve its performance, this form of learning can produce highly accurate predictions compared to the other methods. Unsupervised learning does not provide a correct output for every input and must learn relevant patterns on its own, causing far less predictable outputs. This can make it more difficult to evaluate the model's performance and adjust it accordingly. Additionally, unsupervised learning

algorithms can sometimes produce unexpected and counter intuitive results when it places relevance on unintended features in the data. Reinforcement often requires a lot of data and computation to train. Additionally the network can fail to converge on a solution as it can be difficult to specify appropriate rewards for a given action.

Disadvantageous of supervised learning

Providing the sufficiently labelled data needed for supervised learning is a time consuming and difficult task. Additionally, the trained network may not generalise well to new, unseen data, particularly if the training data was not representative of the overall population. On the other hand, unsupervised learning is able to be implemented on systems where labelling of the training data is not feasible. It is also able to provide insight on much more general patterns within the data which may not be apparent in supervised learning. Taking this a step further, reinforcement learning is more suited to complex problems which might not be viable for the alternatives. Additionally, since reinforcement learning behaves through interactions with it's environment, it is able to learn a more diverse range of behaviours than supervised or unsupervised learning.

2 Information Theory Analysis

2.1 Information Analysis (498/500)

Calculating the entropy of our system enables us to understand the information relationship between layers in the system over learning. The implemented backpropagation algorithm acts as a markov chain, as the output layer can only learn the information available in the input layer through the hidden layer. A representation of these relationships is as shown:

$$\mathrm{X} \rightarrow \mathrm{Y} \rightarrow \mathrm{Z}$$

X is the image label. Y is a binary discrete representation for activity in the hidden layer using the median of the sigmoid function (0.5) as a cut-off as it provided less noisy results and clearer trends than using the mean of the hidden layer. Z is the activity in the output layer.

Conditional entropy's $H(Y|X)$ and $H(Z|X)$ represent the remaining information left in a variable given we already know the outcome of another. This shows the remaining information in our hidden layer Y and output layer Z in relation to a given label X. Figure 8 demonstrates changes in $H(Y|X)$ over learning for various hidden layer sizes. There is an initial sharp drop in $H(Y|X)$ corresponding to the quick learning of our system shown in Figure 4a. The significance of this drop depends on the size of the the hidden layer. Smaller sizes experience a bigger drop than larger sizes because a larger hidden layer can be more representative of the 784 pixel input and therefore have more variance in the unique activation per label presented causing a higher $H(Y|X)$.

$H(Z|X)$ in Figure 8 shows Z to be initially full determined by X with a sharp rise before falling with a decreasing gradient over learning. Entropy is a measure of information and not salience, before any learning occurs there is likely to be very little variance in the activity at the output layer causing lower $H(Z|X)$. As the system learns this variance increases rapidly, then as the system becomes more accurate the different categorisations in the output layer become less varied and the gradient declines.

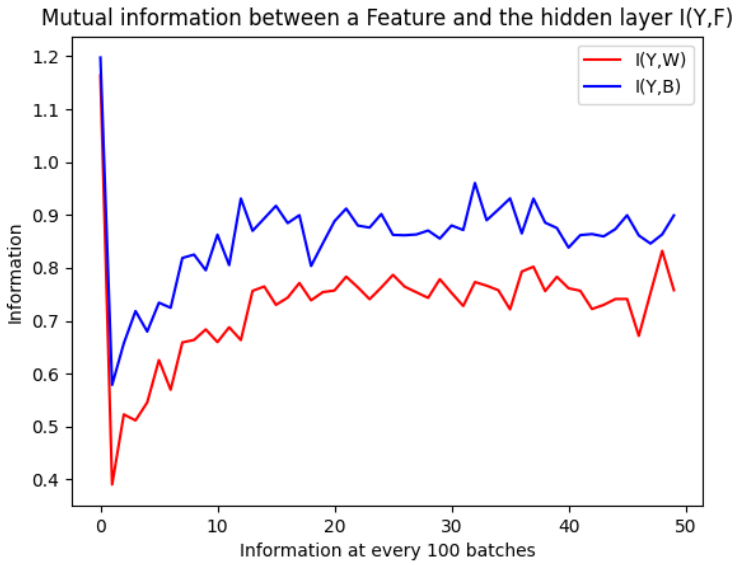
Figure 8: *Information relationships of the system over learning, bs* $= 48$*, tbs* $= 1000$*, lr* $= 0.01$

Mutual information measures the information we can determine from a variable by looking at another variable [10] and therefore $I(X,Y)$ and $I(X,Z)$ represent the mutual dependence between X and the layers Z and Y. The markov chain property $I(X,Y) \geq I(X,Z)$ is presented in Figure 8. Both $I(X,Y)$ and $I(X,Z)$ increase over learning showing that they are both becoming more informative about the input, however, $I(X,Z)$ never has more information about X than $I(X,Y)$. In $I(X,Y)$ larger hidden layers seem give a larger $I(X,Y)$, suggesting they are more informative of the input and could be a reason larger layers provide better learning. The hidden layer size seems to have insignificant effect on $H(Z|X)$ and $I(X,Z)$ over learning.

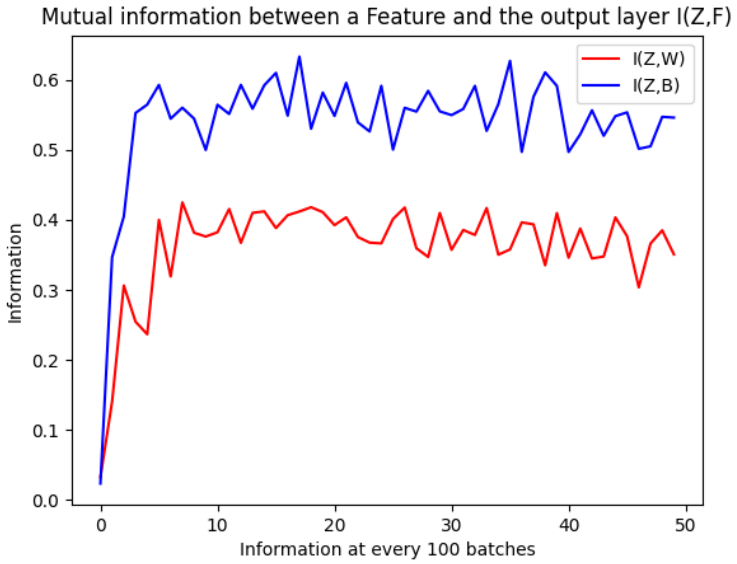
2.2 Information given an image property (240/250)

By considering features of the images and evaluating the mutual information relationship between the layers, we can determine how much information the layers contain about each feature. A hidden layer $H = 16$ is chosen as Figure 8 suggests smaller hidden layer sizes exhibit more distinct changes over learning. Figure 9 evaluates the mutual information relationships between the layers Y and Z with regards to the blackest and whitest quadrants, B and W. Figure 9a shows very marginal difference between the values of $I(Y,W)$ $I(Y,B)$ over learning, suggesting the hidden layer has similar levels of information with regards to each feature. The mutual information in the output layer is more

significant. Figure 9b shows $I(Z,B) \geq I(Z,B)$ throughout learning. This suggests that B holds more information about the output layer than W and is therefore more significant in determining the output. When comparing these figures to the mutual information between the layers and X in Figure 8 it can be determined that both features have lower entropy for both layers than the label. This is likely due to the label being representative of the whole image, rather than one quadrant.



(a) *Mutual Information analysis between the hidden layer and a feature W, B*

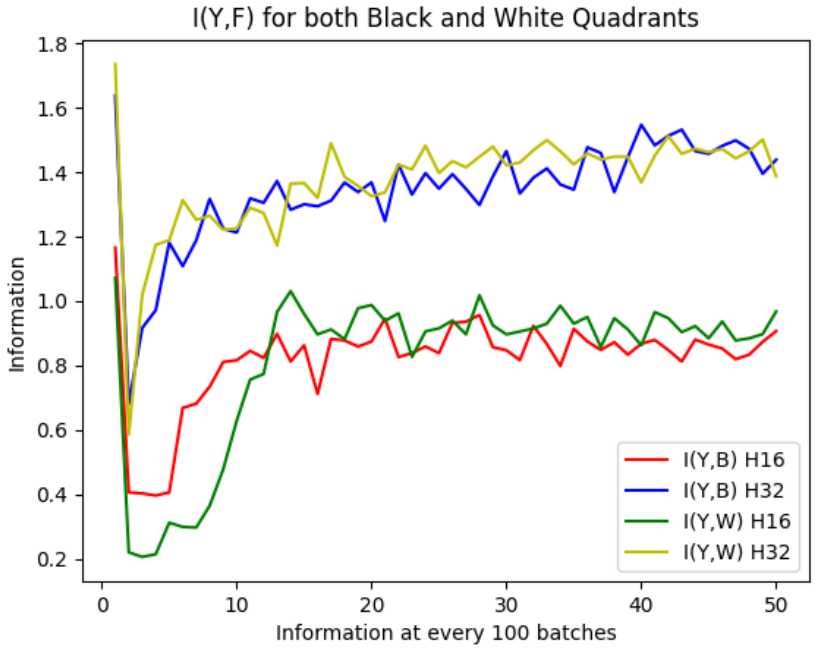


(b) *Mutual Information analysis between the output layer and a feature W, B*

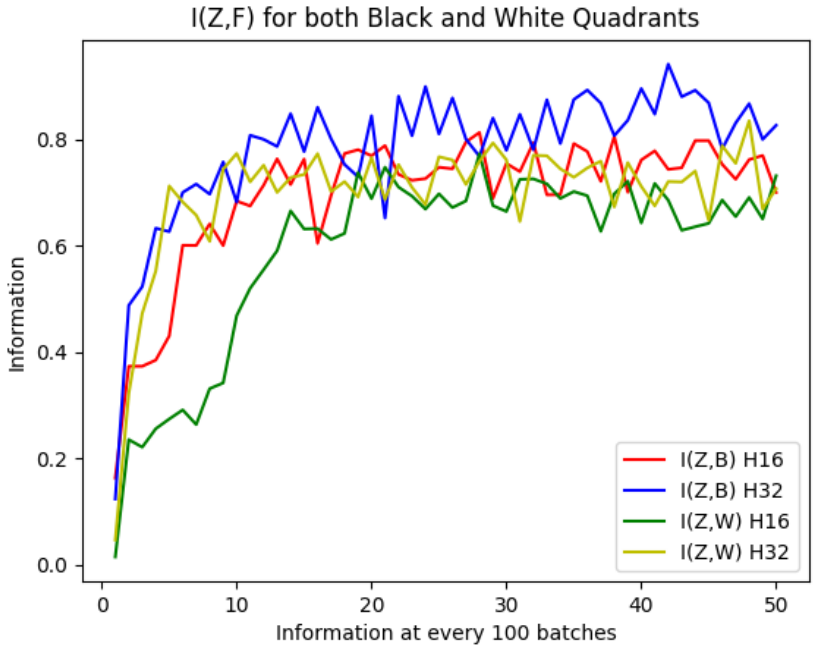
Figure 9: *Mutual information analysis of image features* $H = 16$, $bs = 48$, $tbs = 1000$, $lr = 0.01$

2.3 Autoencoder (248/250)

An autoencoder is a type of neural network that is used to learn a compressed representation of data. This is done by training the autoencoder to reconstruct the original input from the learned encoding which is represented by the hidden layer in our neural network. Applying the autoencoder to our network requires a balanced output and input size. Additionally, we need to calculate the loss function by comparing the output of the network to the inputs. Figure 10 represents the mutual information of the hidden and output layers for the features B and W for hidden layer sizes $H = 16$ and $H = 32$. The autoencoder reduced the difference in mutual information at the output layer and hidden layer, suggesting they hold comparable information with regards to both features. It can be seen that the largest factor on mutual entropy in the hidden layer, is the size of the hidden layer. Larger hidden layers increase the amount of mutual information, this is likely due to the fact that a larger hidden layer is able to be more representative of the input data (i.e. is compressed less), therefore increasing the amount of information the hidden layer has about each feature and increasing mutual entropy.



(a) *Mutual Information analysis between the hidden layer and a feature W, B in an autoencoder.*



(b) *Mutual Information analysis between the output layer and a feature W, B in an autoencoder.*

Figure 10: *Autoencode mutual information analysis:* $bs = 48$, $tbs = 1000$, $lr = 0.01$

References

[1] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[2] Zalandoresearch, "Zalandoresearch/fashion-mnist: A mnist-like fashion product database. benchmark." [Online]. Available: https://github.com/zalandoresearch/fashion-mnist

[3] M. Roodschild, J. Gotay Sardiñas, and A. Will, "A new approach for the vanishing gradient problem on sigmoid activation," *Progress in Artificial Intelligence*, vol. 9, no. 4, pp. 351–360, 2020.

[4] B. A. Richards and T. P. Lillicrap, "Dendritic solutions to the credit assignment problem," *Current Opinion in Neurobiology*, vol. 54, pp. 28–36, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0959438818300485

[5] T. P. Lillicrap, D. Cownden, D. B. Tweed, and C. J. Akerman, "Random synaptic feedback weights support error backpropagation for deep learning," *Nature communications*, vol. 7, no. 1, pp. 1–10, 2016.

[6] W. Greedy, H. W. Zhu, J. Pemberton, J. Mellor, and R. P. Costa, "Single-phase deep learning in cortico-cortical networks," 2022. [Online]. Available: https://arxiv.org/abs/2206.11769

[7] J. Sacramento, R. Ponte Costa, Y. Bengio, and W. Senn, "Dendritic cortical microcircuits approximate the backpropagation algorithm," *Advances in neural information processing systems*, vol. 31, 2018.

[8] T. P. Lillicrap, A. Santoro, L. Marris, C. J. Akerman, and G. Hinton, "Backpropagation and the brain," *Nature Reviews Neuroscience*, vol. 21, no. 6, pp. 335–346, 2020.

[9] B. Mahesh, "Machine learning algorithms-a review," *International Journal of Science and Research (IJSR).[Internet]*, vol. 9, pp. 381–386, 2020.

[10] Wikipedia, "Mutual information — Wikipedia, the free encyclopedia," http://en.wikipedia.org/w/index.php?title=Mutual%20information&oldid=1126376927, 2022, [Online; accessed 09-December-2022].