Modern Hopfield Network with Local Learning Rules for Class Generalization

Shruti Joshi
Department of Electrical and Computer Engineering
University of California San Diego
La Jolla, CA 92093
s4joshi@ucsd.edu

Giri Krishnan
Department of Medicine
University of California San Diego
La Jolla, CA 92093
gkrishnan@health.ucsd.edu

Maxim Bazhenov
Department of Medicine
University of California San Diego
La Jolla, CA 92093
mbazhenov@health.ucsd.edu

Abstract

The Modern Hopfield Network (MHN) model, recently introduced as an extension of Hopfield networks, allows for the memory capacity to scale non-linearly with the size of the network. In previous works, MHNs have been used to store inputs in its connections and reconstruct them from partial inputs. In this work, we examine if MHN can be used for classical classification tasks that require generalization to unseen data from same class. We developed a Modern Hopfield Network based classifier with the number of hidden neurons equal to number of classes in the input data and local learning that is able to perform at the accuracy as MLP on several vision tasks (classification on MNIST, Fashion-MNIST and CIFAR-10). Our approach allows us to perform classification, pattern completion, noise robustness and examining the representation of individual classes within the same network. We identify that temperature determines both accuracy and noise robustness. Overall, in this preliminary report, we propose a simple framework for class generalization using MHN and demonstrates the feasibility of using MHN for machine learning tasks that require generalization.

1 Introduction and Related Works

Hopfield networks [1] are associative memory models with binary neurons and pairwise synaptic connections defined by an energy function. The memory capacity of classical Hopfield networks scales linearly with the number of input features. Introducing higher-order polynomial interactions between neurons can increase the capacity [2] and exponential neuron interactions enable exponential storage capacity [3]. The modern Hopfield network (MHN) developed by [4] uses continuous neuron states and an exponential interaction function, enabling storage of exponentially many patterns. To represent this with pairwise interactions, a two-layer model was proposed by [5]. MHNs have been applied to sequence classification [6] and tabular data [7]. However, the ability of MHNs to learn classes and class generalization has not been extensively evaluated, to our knowledge. One approach for classification with MHNs entails dedicating one hidden unit per training exemplar, irrespective of the class structure present in the data [5, 8]. However, the resulting sizable hidden layer and input-dependence of learned weights limit generalization. We present a modern Hopfield variant amenable to biological local learning that demonstrates class generalization capabilities. Our core contributions are:

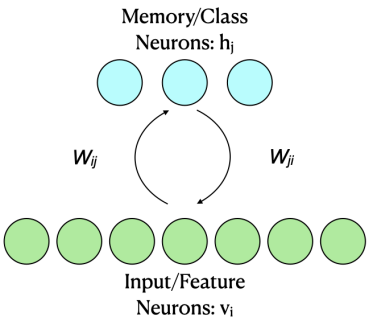
Associative Memory & Hopfield Networks in 2023. NeurIPS 2023 workshop.

- We propose a Modern Hopfield Classifier (MHC) incorporating modified Hebbian learning that demonstrates class generalization capabilities on image classification tasks.
- We show MHC is robust to noise, able to perform pattern completion and can generate internal representation corresponding to a class prototype.
- We find that decaying the inverse temperature parameter over training epochs enhances generalization performance, as quantified by test accuracy.

2 Model description

2.1 Modern Hopfield Classifier

We propose a trainable, continuous valued modern Hopfield model based on the continuous time mathematical formulation of associative memory networks in [5] that can perform class generalization. We call this model the Modern Hopfield Classifier (MHC).



$$\tau_v \frac{dv_i}{dt} = -v_i + \Sigma_{j=1}^{N_h} W_{ij} * f_j \tag{1}$$

$$\tau_h \frac{dh_j}{dt} = -h_j + \Sigma_{i=1}^{N_f} W_{ij} * g_i \tag{2}$$

$$f_j = softmax(\beta * h_j) \qquad g_i = v_i$$

Figure 1: Model of the Modern Hopfield Classifier (reproduced from [5])

The model comprises two layers of neurons (Figure1) - the feature/input neurons (N_f in number) with currents denoted by v_i and the memory/hidden neurons (N_h in number) with currents denoted by h_j connected by a set of symmetric weights $W_{ij} = W_{ji}$. The respective activation functions are $g(\mathbf{v})$ and $f(\mathbf{h})$, with time constants τ_v and τ_h. We use $g(\mathbf{v}) = \mathbf{v}$ and $f(\mathbf{h})$ = softmax($\beta\mathbf{h}$) where β denotes an inverse temperature parameter that controls sharpness of the softmax function. Our formulation during inference resembles the model (type B) described in [5], but differs in that $\mathbf{h}$ evolves dynamically over continuous time rather than instantaneously updating. Another key difference is how the weights are computed; in [5], the weights are set simply as the images which are encoded. In our model, we obtain the weights via training which enables them to capture the class structure present in the data. The dynamics are formalized as a system of coupled nonlinear differential equations (eqns. 1 and 2). We integrate these numerically using the Euler method for 100 steps during both training and inference. Class predictions are obtained by taking the argmax over $\mathbf{h}$ after the 100 integration steps in the inference stage.

2.2 Local Learning Rule

We train the model weights using a modified Hebbian rule inspired by [9]. The synaptic weights W_{ij} are randomly initialized, while the visible neurons v_i are clamped to input data samples. During training, the hidden neurons h_j are initialized randomly and allowed to evolve per eqn2, concurrently with the weight updates computed as:

$$\frac{dW_{ij}}{dt} = \frac{\eta * g_i * (t_j - f_j)}{e^{-|W_{ij}|}} \tag{3}$$

Here f_j = softmax(βh_j) denotes the hidden unit activities, t_j encodes the target distribution from the class labels, η is the learning rate. $|W_{ij}|$ denotes the absolute value of the weight W_{ij}. The denominator acts as a form of weight normalization that prevents uncontrolled growth of the weights during training. Each training image is presented to the network for 100 time steps during which the weights and the state of memory neuron are both allowed to evolve. We train over multiple full passes of the training set (epochs), updating the hyperparameters η (learning rate) and β (inverse

temperature) on a per-epoch basis. The learning rate used for an epoch is $\eta_{epoch} = \eta/(i_{epoch} + 1)$ and the inverse temperature is $\beta_{epoch} = \frac{\beta}{2^{i_{epoch}+1}}$, where η, β are the initial set hyperparameters and i_{epoch} is the index of the current epoch.

3 Results

We evaluated MHC performance on MNIST [10], Fashion-MNIST [], and CIFAR-10 [11] datasets. For MNIST experiments, models were trained on 10,000 random images over 5 epochs. Inference occurred on the held-out 10,000 test instances, integrating dynamics with the torchdiffeq ODE solver (convergence example in Supplementary Figure4). Inputs were preprocessed to standardized pixel intensities via per-pixel $(x - \mu)/\sigma$ normalization using aggregate dataset statistics. This ensured homogeneous feature scales.

3.1 Performance of MHC on classification tasks

The output class for a given input is determined by initializing the visible neurons $\mathbf{v}$ to input test images and the hidden neurons $\mathbf{h}$ to small random values. The coupled neural dynamics given by equations 1 and 2 are numerically integrated for 100 steps to allow the states $\mathbf{v}$ and $\mathbf{h}$ to evolve. The most active hidden neuron is taken as the predicted label $\hat{y}$. Classification accuracy is computed by comparing $\hat{y}$ to the true label y across all test examples.

The MHC model with a single hidden layer achieved approximately 90% classification accuracy on the MNIST dataset. For comparison, we trained a 2-layer multilayer perceptron (MLP) model having identical neuron counts per layer over 50 epochs on the same MNIST training set. In this preliminary version containing only a single hidden layer, the MHC demonstrated comparable performance to the MLP over multiple datasets, as summarized in Table 1.

Model	MNIST	Fashion-MNIST	CIFAR-10
MHC (with temperature scheduling)	90.13%	82.8%	47.31%
MHC (without temperature scheduling)	85.2%	77.13%	41.68%
2-layer MLP	90.67%	83.0%	48.73%

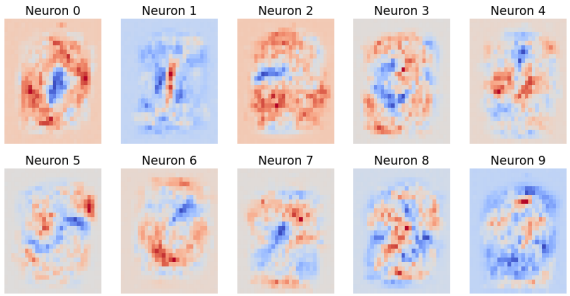
Table 1: Performance (Classification accuracy) of the MHC model on classification tasks

3.2 Class specific representations generated by the MHC model

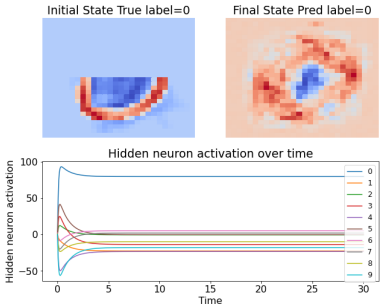
The images generated for each class in Figure2a depict the final states that the feature neurons converge to during inference. These representations are obtained by initializing said neurons randomly, clamping the respective class-specific hidden unit, and integrating the dynamics (eqns. 1 and 2). As such, they approximate the prototypes learned in the model for each category. The MHC learns these general representations during training, rather than memorizing particular instances of images. This facilitates generalization - allowing accurate predictions on novel test data. Additionally, learning these class prototypes within the model parameters enables pattern completion. Upon presenting a corrupted or incomplete input image, the dynamics evolve the state of the feature neurons (v_i) toward the nearest encoded prototype, effectively filling in missing information. Figure2b shows the incomplete image of the digit 0 (initial state of feature neurons) and the image reconstructed by the MHC (final state of feature neurons). The completion of all the digits to their nearest prototype can be seen in Supplementary Figure6. Unlike feedforward networks, capabilities like prototypical pattern generation and completion are result of attractor characteristics of MHC, a distinguishing feature.

3.3 Effects of temperature on classification performance in the presence of noise

The temperature parameter (β) determines the sharpness of the softmax function and hence the accuracy of classification. In Figure3a, it can be seen that the the accuracy at very low β (high temperature) is at chance and as β increases (temperature decreases), the classification accuracy increases and then saturates at around 90% for the unperturbed input and at around 53% for the input with noise added ($\sigma = 4.0$).

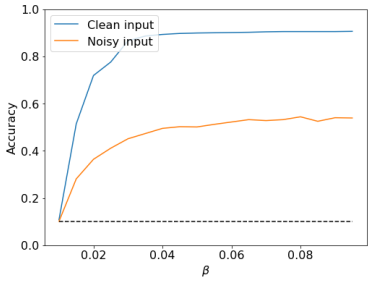


(a) Representations generated by activating each class specific hidden neuron after MHC is trained on MNIST

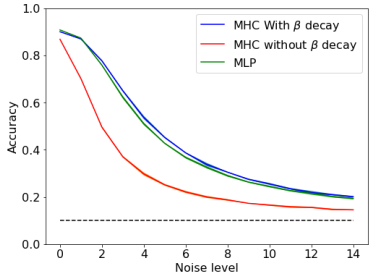


(b) Masked and reconstructed image for the digit 0. The bottom graph shows the evolution of hidden neuron states till convergence.

Decaying β over training epochs exposes the model to diverse response patterns, conferring noise robustness absent when fixing β (Figure3b and Table1). The MHC with β decay marginally exceeds the MLP, while the variant without decay performs substantially worse. The accuracy gap between these MHC models is small on original data yet grows under increasing noise. Optimal task-specific weights emerge via controlled β decay schedules that enable generalized learning.



(a) Effect of β on classification accuracy



(b) Effects of input noise on the accuracy of the MHC model.

4 Discussion and Future Work

In this preliminary work, we propose a novel model of associative memory, trained using a local learning rule, which can perform class generalization. We report results on a small, 2-layer model with just 10 hidden neurons leading to relatively low accuracy compared to state of the art. However, the classification performance is comparable to an equivalent 2-layer MLP. We used the evolution of the coupled dynamical system (weights+hidden neurons) based on local interaction for learning the weights. The time constant of the evolution of weights was much slower than that of the evolution of hidden neuron states, which enabled the attractor dynamics to converge to previously learned classes. Energy Based Models, such as MHN, allow for a larger diversity of responses when the temperature is increased. Consistent with this observation, we show that increasing the temperature after each epoch during learning improves the noise robustness of the model. This allows the model to learn more general features of the class rather than memorizing training images (prevents overfitting to the training set), resulting in increased classification accuracy and robustness to additive noise. Future work will explore storing multiple representations per class, enabled by the rich dynamics of our modern Hopfield variant.

Acknowledgments and Disclosure of Funding

Research reported in this work was supported by the National Institutes of Health (NIH) [1R01DC020892] and the National Science Foundation (NSF) [2209874, 2223839]

References

[1] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79(8):2554–2558, April 1982.

[2] Dmitry Krotov and John J. Hopfield. Dense Associative Memory for Pattern Recognition. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[3] Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299, July 2017. arXiv:1702.01929 [math].

[4] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, David Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield Networks is All You Need. *arXiv:2008.02217 [cs, stat]*, April 2021. arXiv: 2008.02217.

[5] Dmitry Krotov and John Hopfield. Large Associative Memory Problem in Neurobiology and Machine Learning. *arXiv:2008.06996 [cond-mat, q-bio, stat]*, April 2021. arXiv: 2008.06996.

[6] Michael Widrich, Bernhard Schäfl, Milena Pavlović, Hubert Ramsauer, Lukas Gruber, Markus Holzleitner, Johannes Brandstetter, Geir Kjetil Sandve, Victor Greiff, Sepp Hochreiter, and Günter Klambauer. Modern hopfield networks and attention for immune repertoire classification. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18832–18845. Curran Associates, Inc., 2020.

[7] Toshihiro Ota, Ikuro Sato, Rei Kawakami, Masayuki Tanaka, and Nakamasa Inoue. Learning with partial forgetting in modern hopfield networks. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 6661–6673. PMLR, 25–27 Apr 2023.

[8] Beren Millidge, Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, and Rafal Bogacz. Universal hopfield networks: A general framework for single-shot associative memory models. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15561–15583. PMLR, 17–23 Jul 2022.

[9] Jan Melchior and Laurenz Wiskott. Hebbian-Descent, May 2019. arXiv:1905.10585 [cs, stat].

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

[11] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research).

[12] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural Ordinary Differential Equations, December 2019. arXiv:1806.07366 [cs, stat].

[13] Beren Millidge, Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, and Rafal Bogacz. Universal Hopfield Networks: A General Framework for Single-Shot Associative Memory Models, June 2022. arXiv:2202.04557 [cs].

[14] Robert Geirhos, Carlos R. M. Temme, Jonas Rauber, Heiko H. Schütt, Matthias Bethge, and Felix A. Wichmann. Generalisation in humans and deep neural networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[15] Gabriele Lagani, Fabrizio Falchi, Claudio Gennaro, and Giuseppe Amato. Synaptic Plasticity Models and Bio-Inspired Unsupervised Deep Learning: A Survey, July 2023. arXiv:2307.16236 [cs].

[16] Javier R. Movellan. Contrastive Hebbian Learning in the Continuous Hopfield Model. In David S. Touretzky, Jeffrey L. Elman, Terrence J. Sejnowski, and Geoffrey E. Hinton, editors, *Connectionist Models*, pages 10–17. Morgan Kaufmann, January 1991.

[17] Dmitry Krotov and John J. Hopfield. Dense Associative Memory is Robust to Adversarial Inputs. *Neural Computation*, 30(12):3151–3167, December 2018. arXiv:1701.00939 [cs, q-bio, stat].

[18] Dmitry Krotov. Hierarchical Associative Memory, July 2021. arXiv:2107.06446 [cond-mat, q-bio, stat].

[19] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and Fujie Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[20] Dmitry Krotov and John J. Hopfield. Unsupervised learning by competing hidden units. *Proceedings of the National Academy of Sciences*, 116(16):7723–7731, 2019.

[21] Hamza Tahir Chaudhry, Jacob A. Zavatone-Veth, Dmitry Krotov, and Cengiz Pehlevan. Long sequence hopfield memory, 2023.

Supplementary Materials

Energy Function of the trained MHC

The energy function of the trained MHC model can be derived from [5]. The energy function for the general model is:

$$E(t) = [\sum_{i=1}^{N_f} v_i g_i - L_v] + [\sum_{j=1}^{N_h} h_j f_j - L_h] - \sum_{i,j} v_i W_i j h_j \tag{4}$$

$$g_i = \frac{\partial L_v}{\partial v_i} \qquad f_j = \frac{\partial L_h}{\partial h_j} \tag{5}$$

where the first term corresponds to the layer of visible/feature neurons, the second term to the layer of hidden/memory neurons and the last term corresponds to the interaction term between the two layers of neurons. Since we have used $f_j = softmax(h_j)$ and $g_i = v_i$, L_v, L_h are

$$L_v = \frac{1}{2}\sum_i v_i^2 \qquad L_h = log(\sum_j e^{h_j}) \tag{6}$$

Algorithm for training MHC

Algorithm 1 Training the MHC model using local learning rule

initialise W with random weights
$t = array(0, 100)$
for *n_epoch = 1 to* num_epochs **do**
  $\eta_{epoch} = \frac{\eta}{n_{epoch}+1}$
  $\beta_{epoch} = \frac{\beta}{2^{n_{epoch}+1}}$ **for** *i = 1 to* N_{imgs} **do**
    get image, label
    initialise target, t_j = generate_zero_array(size=N_h)
    initialise h_{init} = generate_random_array(size=N_h)
    v_{init} = flatten(image)
    t_j[label] = 1.0
    update W according to eqn 3 for t time period
  end
end

The number of neurons in the feature layers is N_f and in the memory layer is N_h. The time constants of the feature and memory neurons are $\tau_f = 1.0$ and $\tau_h = 0.1$. Learning rate $\eta = 0.02$ and inverse temperature parameter $\beta = 1$. The state of the MHC model is described by a set of non-linear differential equations 1, 2. We used the euler method for numerical integration for the evolution of differential equations for 100 time steps. During inference, the class corresponding to most active hidden neuron following 100 time steps was taken as its output.

Convergence and evolution of states

During inference, the trained Modern Hopfield Classifier dynamics unfold according to Eqs. 2 and 1, evolving the state over time. The visible neurons $\mathbf{v}$ representing input features are initialized to an example image. The memory neurons $\mathbf{h}$, encoding learned prototypes, are randomly initialized. As seen in Figure4, the $\mathbf{h}$ unit corresponding to the correct class becomes increasingly active, ultimately exceeding all other elements of $\mathbf{h}$. This winning unit denotes the network's classification decision upon convergence. Concurrently, $\mathbf{v}$ evolves to a final state that recovers the stored class prototype closest to the input, thus exhibiting associative recall. The integrated dynamics hence classify by activating the appropriate memory unit, while also reconstructing the canonical representation learned for that category.

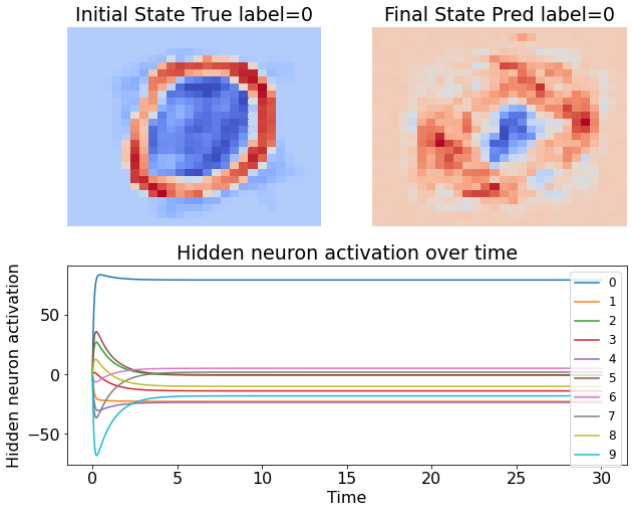
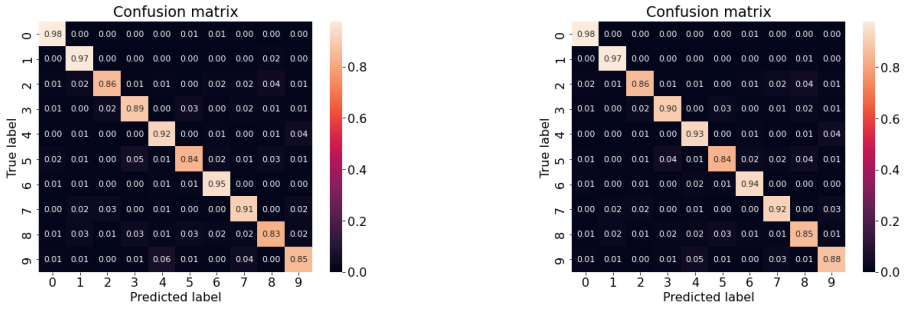


Figure 4: Evolution of the states of MHC model neurons

Confusion matrices for classification on MNIST

In this section, we show the confusion matrices for the classification task performed by the MHC and the 2-layer MLP. In Figure5a and 5b respectively. Both the models were trained on 10,000 training images and tested for 10,000 test images. The MHC was trained for 5 epochs, while the MLP was trained for 30 epochs.



(a) Confusion matrix for MHC

(b) Confusion matrix for the 2-layer MLP

Figure 5: MNIST classification task performance

Pattern completion

In this section, We demonstrate MHC pattern completion capabilities. Figure 7 shows noisy MNIST test images clamped to visible neurons $\mathbf{v}$ converge via MHC dynamics to stored prototypes. Similarly, Supplementary Figure6 presents 40% masked inputs completed by dynamics to learned representations, with model-predicted labels. The obscured "4" completes to a "9", consistent with human perception given ambiguities. The noisy image shows the hidden neuron associated with the correct class being activated, denoting successful classification.

Effect of size of training set on the performance of the MHC

We evaluated MHC and 2-layer MLP classification accuracy versus MNIST training set size, fixing models at 5 training epochs. Figure9 shows accuracies on a held-out test set after fitting each architecture on varying numbers of examples. For larger training sets, MHC and MLP performance is comparable. However, MHC slightly outperforms MLP when trained on fewer examples. This indicates enhanced generalization capacities given limited data, a characteristic well-suited for small or sparse datasets.

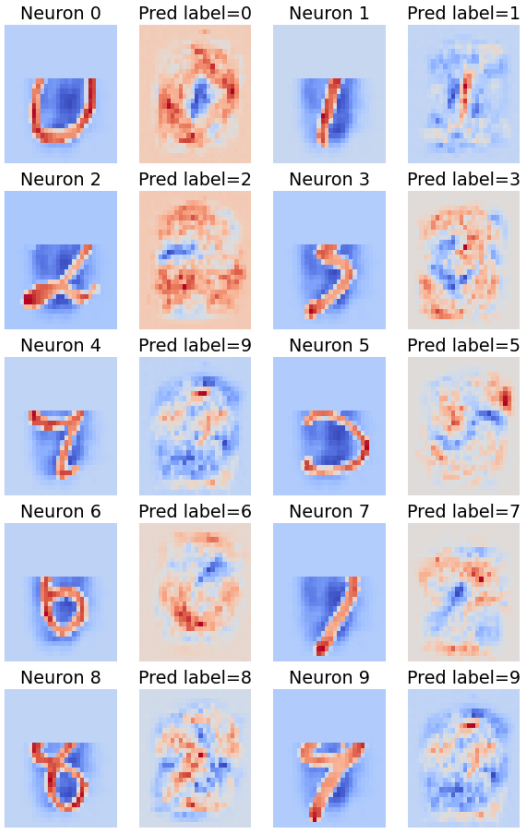


Figure 6: Masked and reconstructed images for each MNIST digit

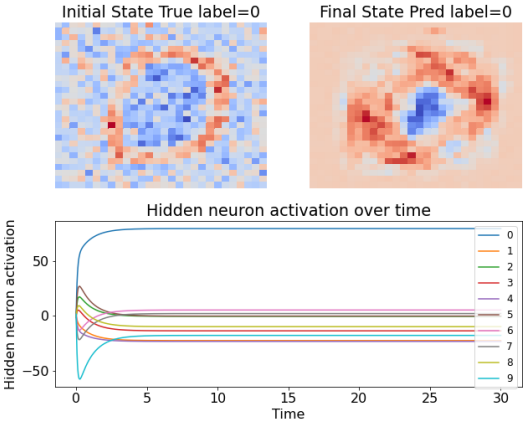


Figure 7: Noisy input

Figure 8: Pattern completion and classification for noisy and masked images

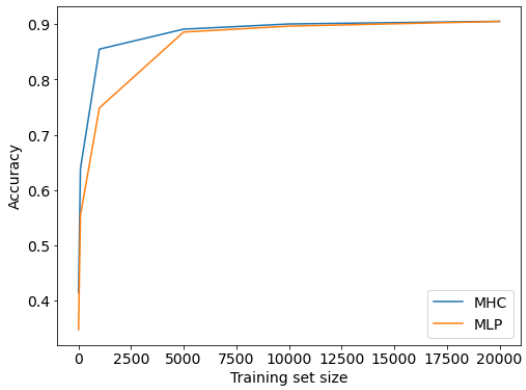


Figure 9: Accuracy vs training data size